Databázové systémy 1

6MDBS1

Ing. Vladimír Přibyl, Ph.D.

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

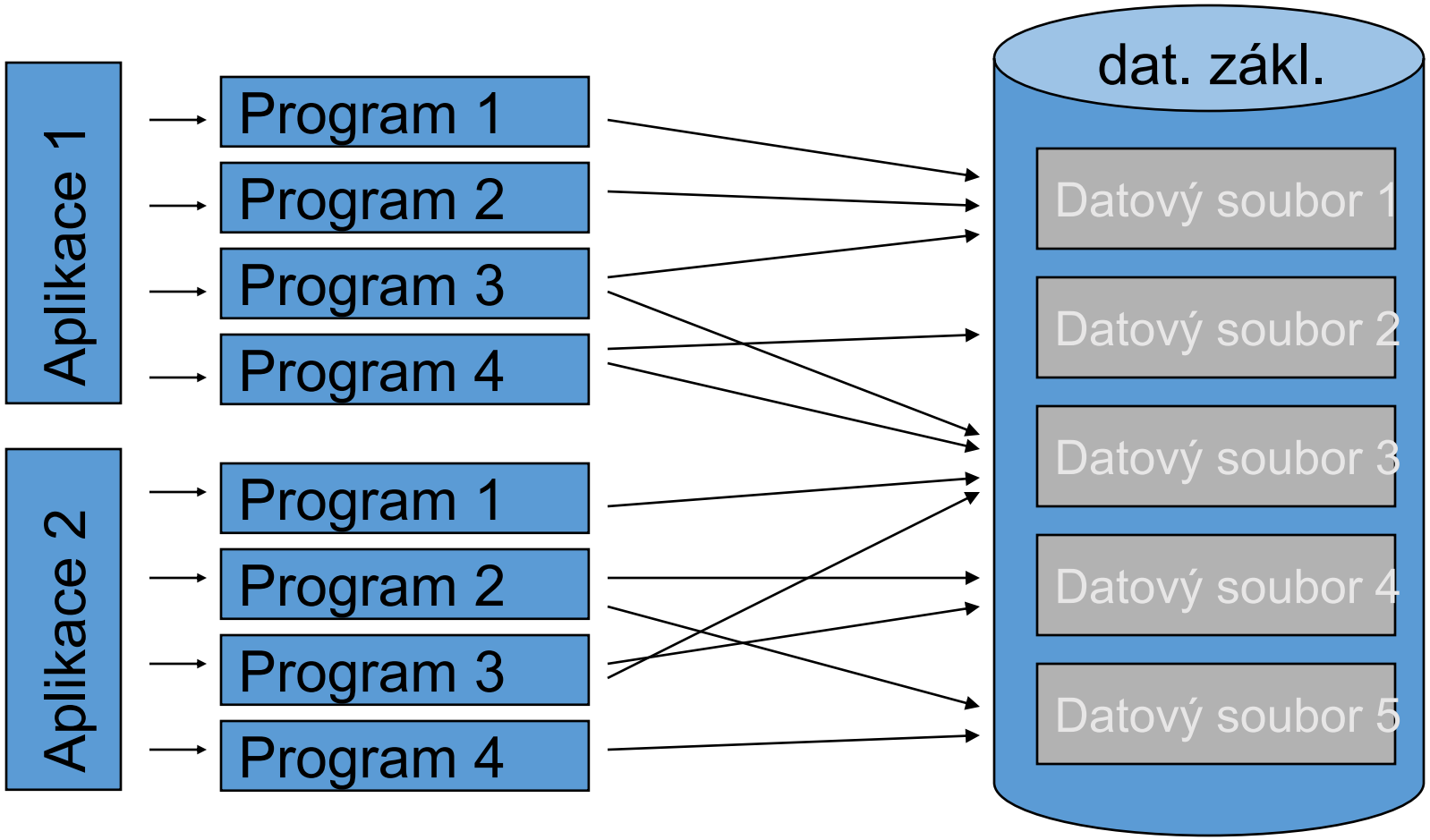
Organizace strukturovaných dat a vyhledávání v nich

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Souborová koncepce dat

- typická především pro minulá desetiletí
- využívá pevně strukturovaných vět
  - ***položka - věta - soubor - datová základna***
- jedná se o éru programovacích jazyků jako FORTRAN,ALGOL, COBOL,...
- datové soubory jsou sdíleny více aplikacemi

Souborová koncepce dat



EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Souborová koncepce
přístup k datům

- základní přístupová jednotka je věta
- věta je v rámci souboru identifikována pořadím, nebo primárním klíčem ***(obvykle jedna z položek, která je jedinečná v rámci celého souboru)***
- základní přístup k větám může být:
  - ***sekvenční***
  - ***přímý***

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Souborová koncepce
nevýhody

- datová redundance
- těsná závislost programů a dat
- obtíže s flexibilitou
- obtíže s ochranou dat
- obtíže se sdílením dat více uživateli

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

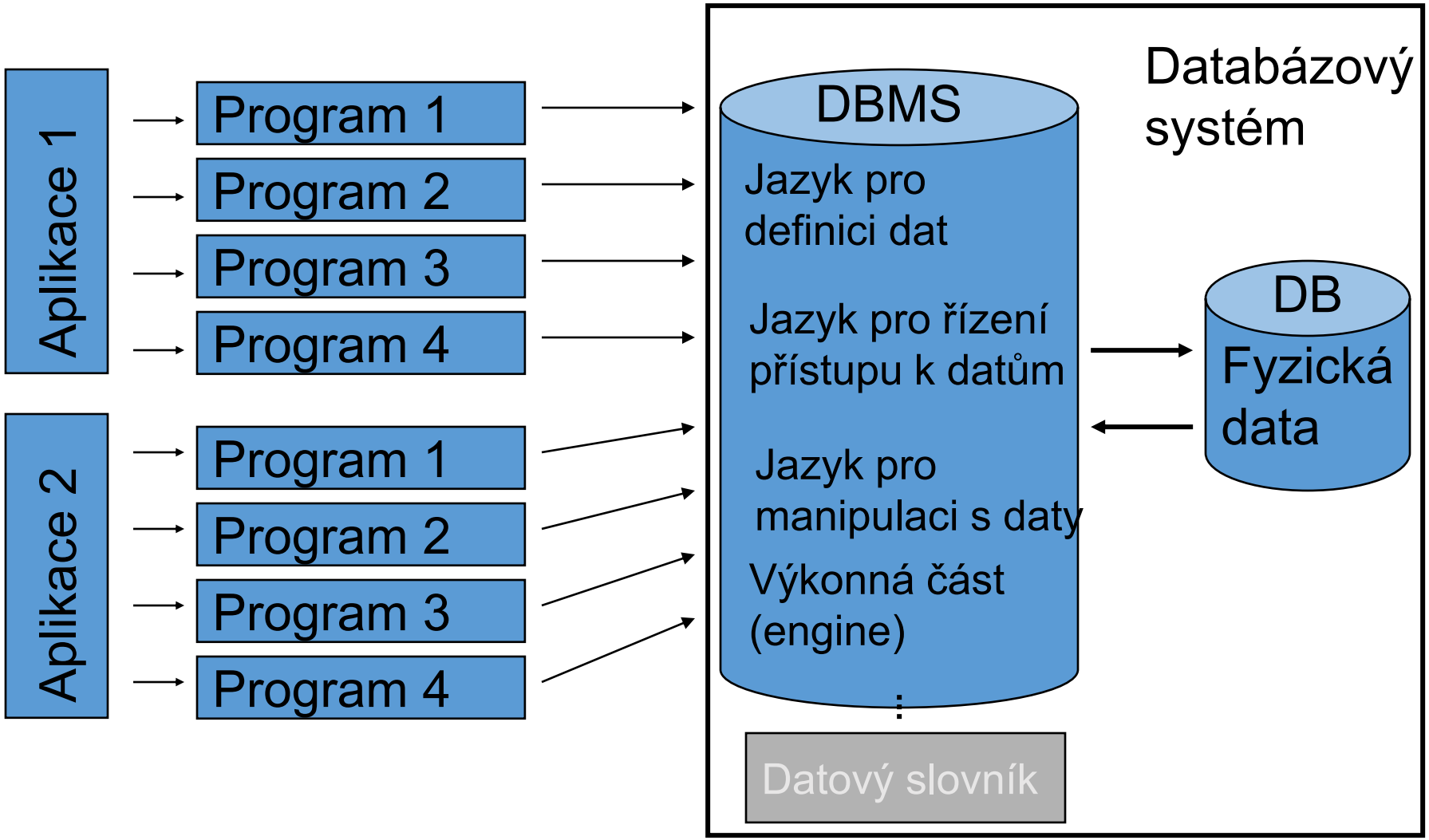
Databázová koncepce

- databáze - ***soubory dat, které slouží více aplikacím, jsou v nich minimalizovány redundance a existuje vhodně centralizovaná správa těchto dat.***
- základním pilířem této koncepce je programový systém umožňující práci s databází:

  DBMS (*Database Management System*)

  resp. **SŘBD** (*Systém Řízení Báze Dat*)

Databázová koncepce dat



EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Komponenty SŘBD

- Databázový stroj (engine, výkonná část)
- jazyk pro definici dat (DDL)
- jazyk pro manipulaci s daty (***DML,QBE***)
- jazyk pro řízení přístupů uživatelů k datům (DCL)
- formátovací jazyk
- datový slovník

Funkce DBMS

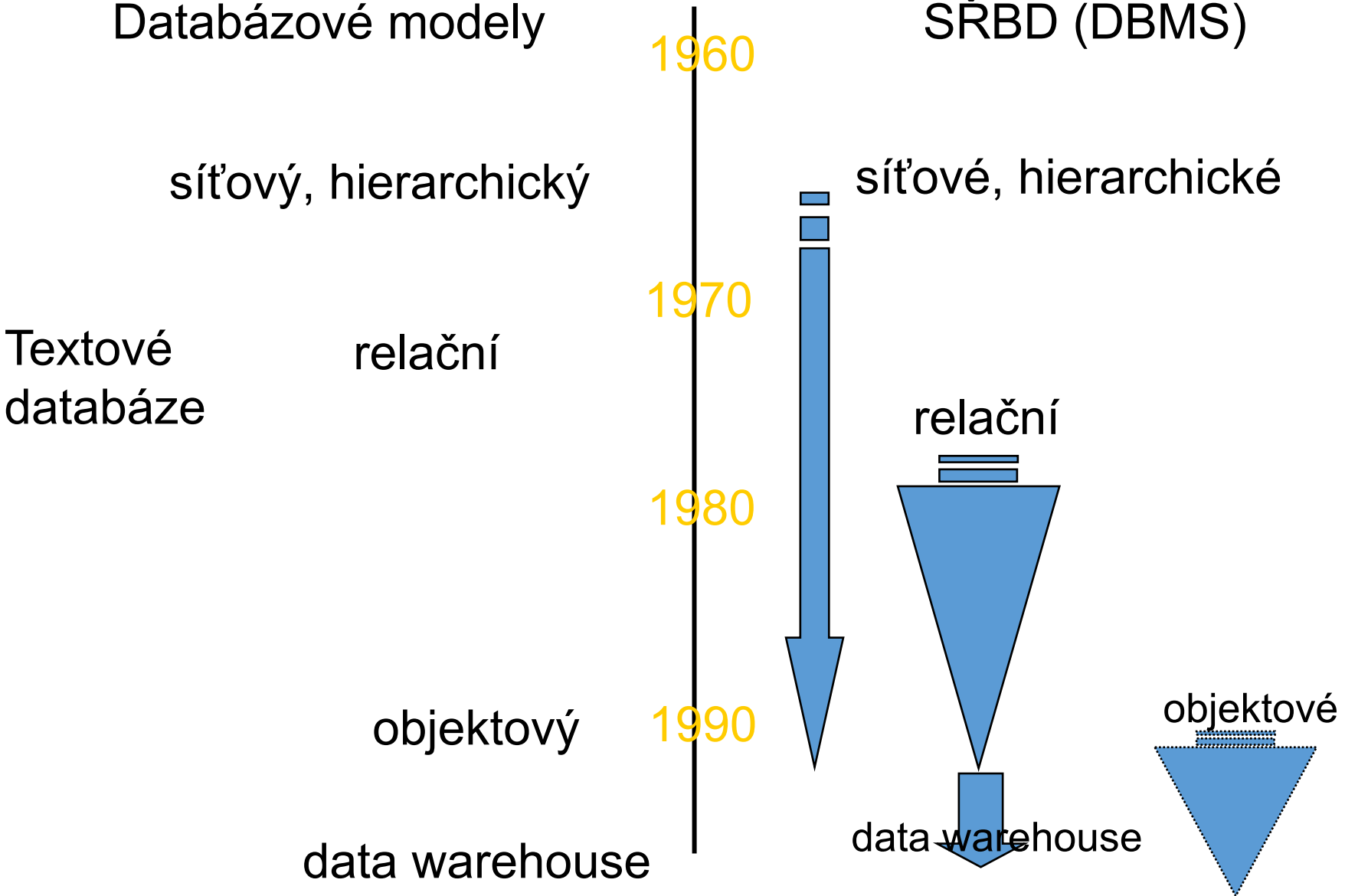
- Vytváření struktur pro ukládání dat, podpůrných struktur
- Údržba databázových struktur
- Čtení dat
- Úpravy dat (vkládání , aktualizace, odstranění)
- Kontrola integrity
- Kontrola souběžnosti
- Zajištění bezpečnosti, zálohování, obnovení

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Typy databázových systémů

- hierarchické
  - historicky nejstarší
  - stromová struktura
- síťové
  - rozšíření o vazby mimo strom
- relační
  - v současnosti nejrozšířenější
- Objektové

Vývoj databázových systémů



EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Data v databázových systémech

- Rozlišujeme zde:
  - **Datové typy**
    - typ obsahu, způsob kódování, způsob práce
  - **Datové struktury**
    - z hlediska vztahů k jiným datům
    - hledisko struktury, tedy formálního uspořádání

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Datové typy

- **vestavěné datové typy**
  - základní datové typy, pro které má databázový systém vlastní prostředky na správu
  - v jednotlivých realizacích databázových systémů se mohou mírně lišit (názvem, rozsahem hodnot apod.)
    - pro relační DBS jsou datové typy do značné míry definovány v rámci jazyka SQL
  - **číselné datové typy**
    - liší se především rozsahem přesností, tedy délkou použitého kódu
      - reálná čísla
      - celá
    - speciální číselné formáty
      - Datum/čas
      - finanční (měnové) – zobrazují se 2 desetinná místa
  - **boolean (logické)**
  - **textové hodnoty** (znakové řetězce – písmena, číslice, symboly)
  - **pointery** (ukazatele)

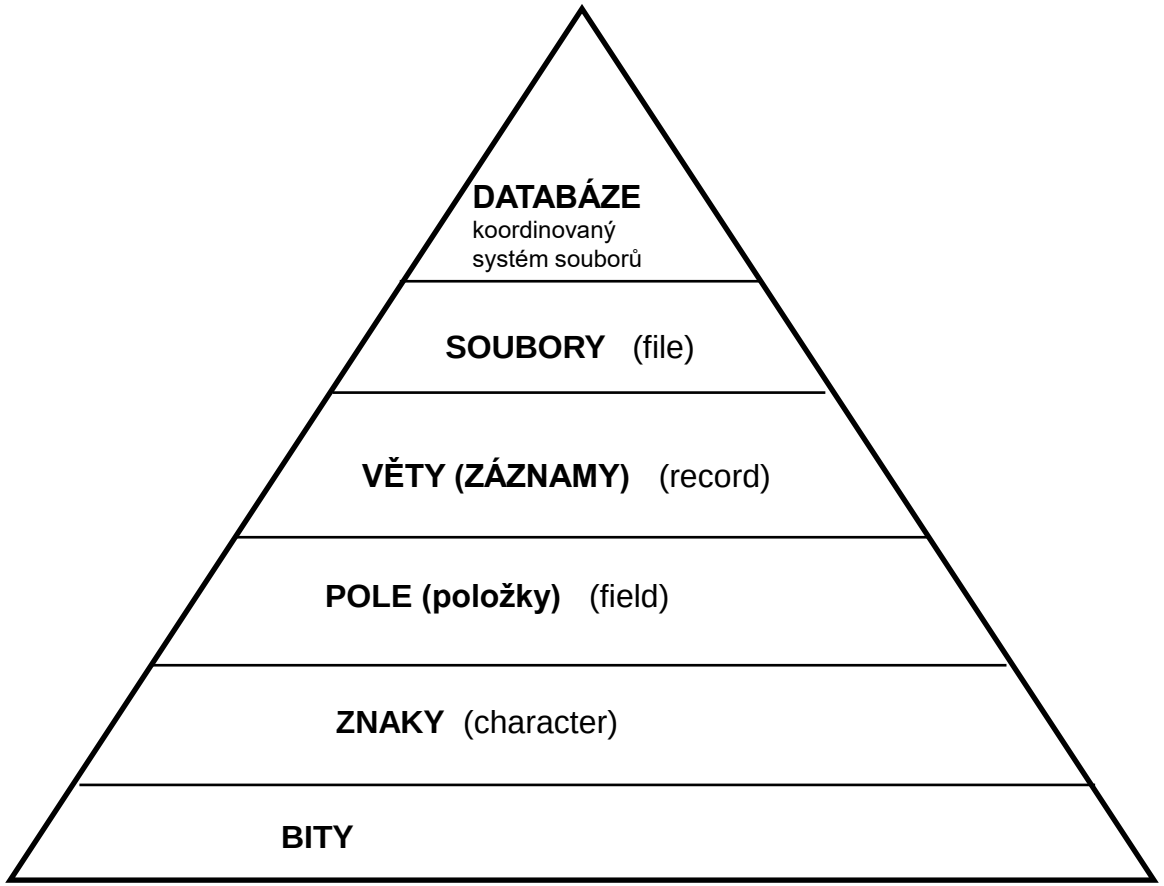
EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Datové typy

- **další datové typy obvykle zahrnované do kategorie BLOB (binary large objects)**
  - Množina bitů s obsahem a strukturou neznámou pro databázový systém. Tím se liší od vestavěných datových typů, neboť s nimi nelze provádět databázové operace (vytvářet indexy, vyhledávat podle nich, porovnávat, třídit).
  - Patří sem
    - **grafické soubory**
    - **zvukové záznamy**
    - **multimediální data**

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Datové struktury



EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Datové struktury

- Zejména se jedná o způsob skládání vyšších struktur z elementárních prvků.
- Podle tohoto přístupu pak rozlišujeme různé typy databázových systémů.
  - lineární (sekvenční) struktura
  - stromová (hierarchická) struktura
  - síťová struktura
  - **relační struktura**
  - objektově orientovaná struktura
- Z hlediska tohoto předmětu nás bude zajímat především výše označena relační struktura
  - nicméně v souvislosti s Big Data se znovu využívají i další struktury

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Technická (fyzická) realizace stromových a síťových struktur

- **pointer – ukazatel**
  - součást záznamu v podobě speciální položky (speciální datový typ), pevně spojující související prvky
  - explicitní odkaz od jednoho záznamu ke druhému
  - jednosměrný
  - obousměrný (oba záznamy ukazují na sebe navzájem)

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Relační databázové systémy

- veškerá data pojímáme jako matematické relace, t.j. množiny uspořádaných n-tic (Tuple)
  $\{(a_1 , \ldots . , a_n)\}$ kde a_i jsou prvky množin M_1 až M_n

  a_i jsou hodnoty atributů a M_i jsou domény atributů
- relace je tedy v podstatě tabulka s pojmenovanými sloupci
  - jednotlivé sloupce = atributy
  - řádky tabulky = záznamy, relace
- relací může být v databázi libovolný počet a mohou být vzájemně propojeny vazbami

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Typy relací

- entitní relace
  - n-tice atributů popisující vlastní entity(objekty)
- vztahové relace
  - n-tice atributů tvořících klíče entit vstupujících do vztahu
  - n-tice atributů popisujících samotný vztah

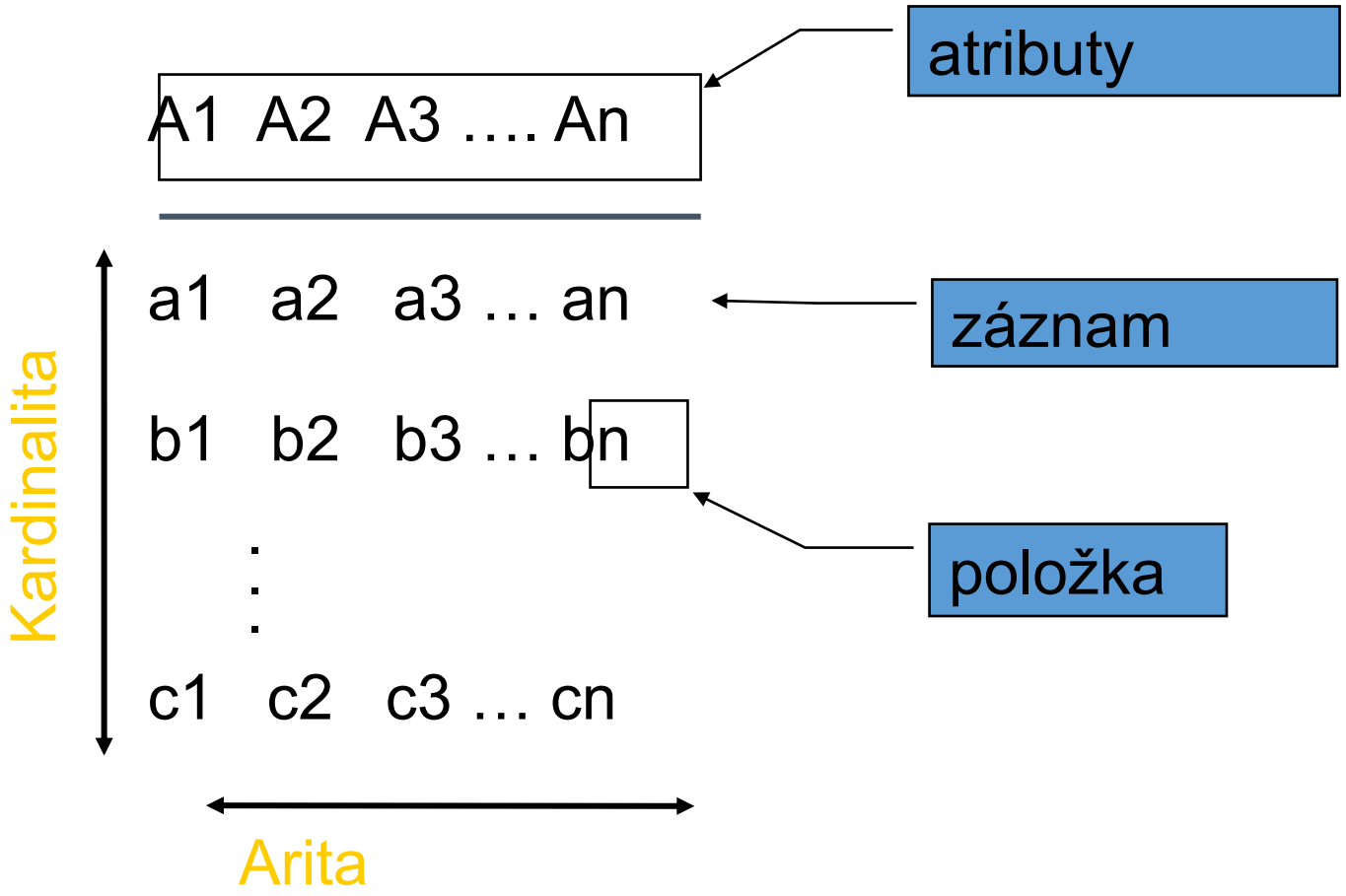
EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Relační model - pojmy

- Relační schéma
  - seznam atributů a dalších údajů o struktuře relace (klíče, omezení, domény ..)
- Instance relace
  - konkrétní množina řádků pro dané relační schéma
- Databázové schéma
  - kolekce relačních schémat

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Relační model - pojmy



EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Relační algebra

- prostředek pro práci s relacemi
- významné operace:
  - projekce (výběr atributů)
  - selekce (výběr záznamů)
  - spojení (spojení relací do nové relace)

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Relační algebra – přehled symbolů

- σ Selekce
- π Projekce

} unární

- X kartézský součin
- $\cup$ Sjednocení
- - Rozdíl

} binární

} základní

- $\cap$ Průnik
- $\triangleright\!\!\triangleleft_\theta$ Theta- join (spojení)
- $\triangleright\!\!\triangleleft$ Přirozené spojení

} Mohou být definovány pomocí základních

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Selekce

- $R = \sigma_C(S)$
  - Kde R,S jsou relace a C je podmínka zahrnující atributy relace S
  - C má tvar výrazů typu„**atribut operator hodnota“**
    - operátory $<,>,=,\neq,\leq,\geq$
    - výrazy lze spojovat log. operátory
  - **$arita(\sigma_c(R)) = arita(R)$**
  - **$0 \leq kard(\sigma_c(R)) \leq kard(R)$**
- Příklad:
  - prazaci = $\sigma_{mesto = praha}(matrika)$

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Projekce

- $R = \pi_L(S)$
  - Kde R,S jsou relace a L je seznam atributů relace S
  - **$arita(\pi_L(R)) \leq arita(R)$**
  - **$0 \leq kard(\pi_L(R)) \leq kard(R)$**
- Příklad:
  - $\pi_{mesto,ulice,PSC}(matrika)$

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Kartézský součin

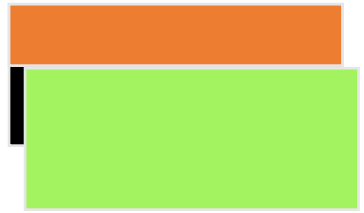
- Kombinuje záznamy obou relací každý s každým
- arita(RxS) = arita(R)+arita(S)
- kard(RxS) = kard(R) . kard(S)

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Sjednocení

- lze aplikovat jen na stejná relační schémata
- množina záznamů nacházející se alespoň v jedné relaci
- **arita(R) = arita(S) = arita(R∪S)**
- **max(kard(R),kard(S)) ≤ kard(R∪S) ≤ kard(R)+kard(S)**

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY



Rozdíl

- lze aplikovat jen na stejná relační schémata
- R – S jsou ty záznamy z R, které nejsou v S
- rozdíl nemá nic společného s vazbami a spojováním

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Průnik

- záznamy, které jsou v R i v S
- dá se definovat jako R-(R-S)
- **arita(R) = arita (S) = arita(R∩S)**
- **0 ≤ kard(R∩S) ≤ min(kard(R),kard(S))**

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Theta spojení

- $R \triangleright_{\theta} \triangleleft S = \sigma_C(R \times S)$
- θ má tvar podmínky ze selekce
- $arita(R \triangleright_{\theta} \triangleleft S) = arita(R) + arita(S)$
- $0 \leq kard(R \triangleright_{\theta} \triangleleft S) \leq kard(R) \,.\, kard(S)$

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Přirozené spojení

- zvláštní případ Theta spojení, kdy všechny stejně se jmenující atributy musí mít stejnou hodnotu
- každá z těchto dvojic sloupců je pak ve výsledku jen jednou

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Pořadí operací

- unární operátory mají přednost před multiplikativními (spojení, kart. součin) a ty ještě před aditivními (sjednocení, rozdíl, průnik)
- vždy je ale lépe používat závorky

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Rozšiřující relační algebra

- přidává funkce důležité pro SQL
  - δ operator eliminace duplikovaných záznamů
  - rozšířená projekce
    - dovoluje výpočty s atributy
  - τ operátor řazení
  - γ operátor seskupování
  - $\triangleright^{o}\triangleleft$ **vnější spojení**

Jazyk SQL

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

SQL

- Structured Query Language
- byl vyvinut pro práci s relačními databázemi
- jedná se o tzv. 4GL jazyk
- jeho základní forma má tyto významné vlastnosti
  - neprocedurálnost (deklarativnost)
    - nepopisuje jak data získat, ale která chceme
    - existují rozšíření kde toto neplatí striktně (PL/SQL, Transact SQL)
  - relační úplnost
  - neobsahuje prostředky pro formátování výstupů

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

SQL

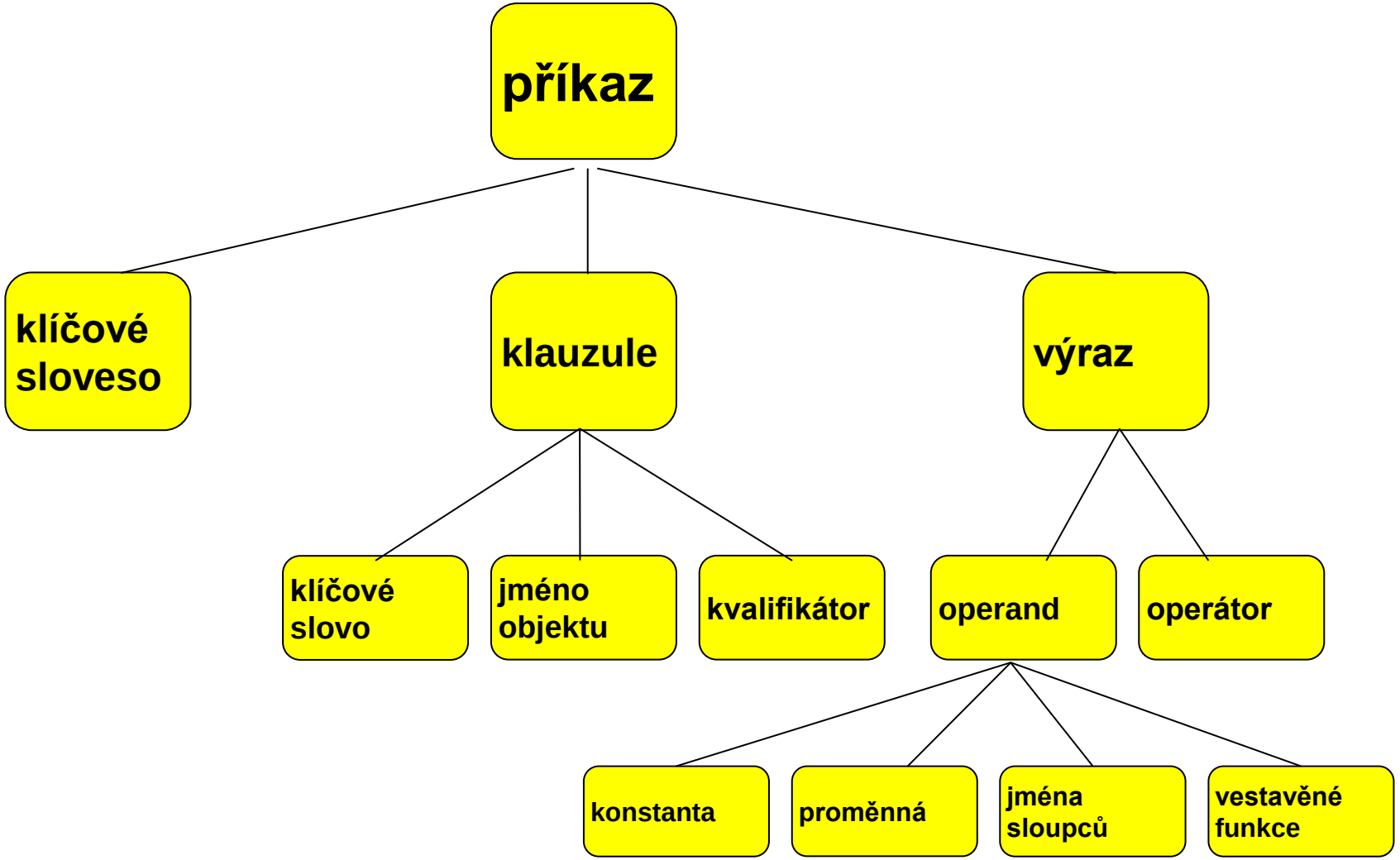
Z hlediska typu operací lze SQL rozdělit na tyto tři základní části:

- jazyk pro definici datových struktur
  (Data Definition Language — DDL)
- jazyk pro manipulaci s daty
  (Data Manipulation Language — DML)
- Jazyk pro řízení přístupů uživatelů k datům
  (Data Control language - DCL)

Historie standardizace SQL

1974-75	IBM ve vývojovém středisku v San José vyvinula 1. verzi SQL (tehdy pod názvem SEQUEL – Structured English Query Language) jako součást prototypových relačních databázových systémů System R a SEQUEL-XRM.
1986	Schválena americká norma ANSI X3.135 (ANSI SQL)
1987	ISO přijala v nezměněné podobě ANSI SQL jako ISO/IEC 9075 (**SQL86**)
1989	Vydán dodatek ISO/IEC 9075 „Integrity Enhancement Feature“, umožňující definovat integritní omezení (**SQL89**)
1992	Schválena norma ISO/IEC 9075:1992 (**SQL92, SQL2**)
od 1992	Příprava **SQL3** (objektová orientace)
1999	Schválena norma ISO/IEC 9075 **SQL:1999** (podmnožina SQL3)
od 2000	SQL4 (nahrazuje SQL3)

Struktura SQL příkazu



EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Struktura SQL příkazu

- **klíčové sloveso:** označuje činnost, kterou daný příkaz provádí (např. SELECT)
- **klauzule:** blíže specifikuje činnost nebo data, s nimiž příkaz pracuje (např. DISTINCT)
- **klíčové slovo:** slovo, kterým klauzule začíná (např. FROM)
- **jméno objektu:** jméno tabulky nebo sloupce
- **kvalifikátor:** jednoznačná specifikace objektu (zpravidla jménem tabulky – např. KNIHY.Autor)
- **výraz:** hodnotu výrazu musí databázový stroj spočítat, pak ji umístí do tabulky výsledků

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Struktura SQL příkazu

- **operand**: část výrazu obsahující data, s kterými se má pracovat
  - konstanta: její hodnota se objeví v každém řádku tabulky výsledků
  - proměnná: symbol, který může nabývat různých hodnot
  - jména sloupců (polí, položek, atributů) – *jméno tabulky.jméno sloupce*
  - vestavěné funkce (agregáty)

- **operátor**:
  - symbol označující operace, které je nutno provést s danými operandy
  - aritmetické
  - logické
  - relační

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Datové typy SQL: 1999

- Textové (znakové)
  - CHARACTER [CHAR] (*n*)
    - textový řetězec délky přesně n
  - CHARACTER VARYING [VARCHAR](*n*)
    - textový řetězec proměnné délky max. n
  - CLOB
- Číselné přesné
  - SMALLINT (4 bajty)
  - INTEGER (8 bajtů)
  - NUMERIC (x,y)
    - reálné číslo
  - DECIMAL(x,*y*)
    - desetinné číslo s pevnou řádovou čárkou

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Datové typy SQL: 1999

- Číselné přibližné – plovoucí řádová čárka
  - FLOAT (x)
  - REAL
    - -3,4E 38 až 3,4E 38
  - DOUBLE PRECISION
    - -1,79E 308 až 1,79E 308
- Datové
  - DATE,TIME, DATETIME,..
- Ostatní
  - BOOLEAN
  - BLOB
- Hodnota NULL
  - Není to 0 ani prázdný řetězec

Data Manipulation Language

- Obsahuje příkazy **SELECT, INSERT, UPDATE a DELETE** pro výběr, vkládání, aktualizaci a mazání dat.
- Jedná se o nejčastěji používanou část jazyka SQL.
- Příkazy z této kategorie jsou uplatňovány samostatně, nebo jsou součástí definice view, uložených procedur a transakcí.

Příkaz SELECT

- slouží pro výběr (čtení) dat z tabulek
- realizuje operace projekce, selekce i spojení
- základní syntaxe:

SELECT [DISTINCT | ALL] {* | <seznam výběru>}
FROM <tabulka> [{, <tabulka>} …]
[**WHERE** <podmínka>]
[**GROUP BY** <specifikace seskupení>]
[**HAVING** <podmínka>]
[**ORDER BY** <specifikace řazení>]

SELECT – pořadí provádění klauzulí

- FROM
- WHERE
- GROUP BY
- HAVING
- výběr výstupních polí
- ORDER BY

Výběr výstupních polí

- syntakticky první část příkazu, hned za slovesem SELECT, případně za kvalifikátory DISTINCT | ALL (nicméně vykonává se téměř na konci)
- může být nahrazen „*“, což znamená výstup všech použitelných polí zdroje
- často bývá spíše ve tvaru
  <odvozený sloupec> [[AS] <název>]
  [{, <odvozený sloupec> [[AS] <název>]}]
- **<odvozený sloupec>** je nejčastěji některý ze sloupců zdroje dat, ale může to být i nějaký výraz, který obsahuje názvy sloupců, konstanty, funkce a operátory
- Kvalifikátory DISTINCT nebo ALL jsou nepovinné. Pokud nebudou zadány, předpokládá se ALL. **DISTINCT** znamená výběr jen různých kombinací hodnot vybraných polí (bez duplicit)

Výběr výstupních polí

- Příklady:

SELECT * …

SELECT **Jmeno**, **Prijmeni**, **datNar** AS DatumNarozeni, …

SELECT **Jmeno** & „ “ & **Prijmeni** AS CeleJmeno, …

SELECT **cena** * 1.2 AS CenaSDPH, …

SELECT AVG(**cena)** AS PrumernaCena, …

SELECT DISTINCT Jmeno, …
výpis jen různých jmen

Klauzule FROM

- jediná povinná klauzule
- používá se pro definici zdroje dat pro dotaz
- vykonává se tedy jako první
- Nejjednodušší varianta zdroje je jedna tabulka. Ostatní možnosti viz „spojení tabulek“
- Příklad:

```
SELECT * FROM Osoby;
SELECT Jmeno, Prijmeni, datNar AS DatumNarozeni FROM Osoby;
```

Klauzule WHERE

- slouží pro definici podmínek výběru záznamů (řádků) a není povinná, nicméně je velmi častá
- provádí se hned po klauzuli FROM
- sestává z jednoho nebo kombinace predikátů (výrazů o jejichž pravdivosti se dá rozhodnout)
  - Základní operátory =, <, >, <=, >=, <>
  - logické spojky AND, OR a operátor negace NOT
  - nalezení hodnot NULL – predikát IS NULL (opak IS NOT NULL)
  - Nalezení podobných hodnot – predikát s LIKE (nazev LIKE „*matematika*“)

Klauzule WHERE – odkazy na další zdroje dat

- přímé srovnání s výsledkem jiného dotazu
  - jen pokud **dotaz vytvoří jednu hodnotu**
  - … WHERE rok **=** (SELECT rok FROM …)
- predikát s **IN**
  - pravdivý, pokud se hodnota nachází v množině
  - … WHERE rok IN (2020, 2019, 2018)
  - … WHERE rok IN (SELECT rok FROM …)
  - … WHERE rok NOT IN (SELECT rok FROM …)
- predikát s **EXISTS**
  - pravdivý, pokud je množina neprázdná
  - neporovnává s hodnotou pole
  - … WHERE EXISTS (SELECT * FROM Osoby WHERE Prijmeni = „Novák“)
  - … WHERE NOT EXISTS (SELECT * FROM Osoby WHERE Prijmeni = „Novák“)

Klauzule WHERE – odkazy na další zdroje dat

- Kvantifikace porovnání s množinou
  - **SOME** nebo **ANY**
    - predikát je splněn, pokud je splněna podmínka alespoň v jednom případě
    - … WHERE body < ANY (SELECT body FROM Hodnoceni WHERE KruhID = 10)
  - **ALL**
    - predikát je splněn, pokud je splněna podmínka ve všech případech
    - … WHERE body < ALL (SELECT body FROM Hodnoceni WHERE KruhID = 10)

Klauzule ORDER BY

- slouží pro řazení
- vykonává se jako poslední část příkazu, až po výběru výstupních polí
- není samozřejmě povinná
- Definuje se výčtem sloupců (odvozených sloupců) s určením způsobu řazení pro každý z nich. Priorita řazení je podle pořadí výčtu.
- Způsob řazení se definuje klíčovými slovy
  - ASC – vzestupné řazení (nepovinné, default)
  - DESC – sestupné řazení
- Příklad:
  … **ORDER BY** body **DESC**, Prijmeni, Jmeno

Klauzule GROUP BY

- používá se k seskupování, ve většině případů za účelem sumarizace
- provádí se po klauzuli WHERE, pokud je v příkazu přítomná
- Syntaxe:

  GROUP BY <odvozený sloupec> [{, <odvozený sloupec>} …] |
  {**ROLLUP** | **CUBE**} (<odvozený sloupec> [{, <odvozený sloupec>} …])

- Příklad:

  … **GROUP BY** pobocka, rok …
  … **GROUP BY ROLLUP** pobocka, rok …
  … **GROUP BY CUBE** pobocka, rok …

Klauzule GROUP BY

- přítomnost klauzule v příkazu ovlivňuje významně možnosti definice výstupních polí
  - Ve výstupních polích mohou být **přímo jen pole, které jsou součástí definice seskupování**.
  - **Ostatní** pole tam mohou být jen **ve spojení s nějakou agregační funkcí**.
    - COUNT
    - SUM
    - AVG
    - MIN
    - MAX
    - další podle implementace (v Accessu např. FIRST, LAST, STDEV, VAR)
- Pozor, i pokud je v definici výstupních polí použit alias (.. AS …), není možné jej použít v definici seskupení. (zejména důležité u odvozených polí)

Klauzule HAVING

- podobně jako klauzule WHERE slouží pro definici podmínek výběru
- může být v příkazu nezávisle na klauzuli WHERE i GROUP BY, nicméně v drtivé většině případů je spojena s klauzulí GROUP BY a slouží pro aplikaci podmínek na již seskupené záznamy s vypočtenými agregačními funkcemi.
- Syntaxi má stejnou jako WHERE, ale v predikátech se může odkazovat jen na pole použitá v klauzuli GROUP BY, nebo s aplikovanou agregační funkcí.
- Odkazy na aliasy z definice výstupních polí nejsou možné

```
SELECT rok, sum(tržba) AS Celkova_Trzba
… GROUP BY rok
HAVING rok > 2010 AND sum(trzba) > 1000000 …
```

Spojování tabulek

- děje se v rámci klauzule FROM
- spojovat je samozřejmě možné i několik tabulek, ale obvykle jde o tabulky, které mají mezi sebou v rámci databázového modelu definovaný vztah
- součástí spojení může samozřejmě být i výsledek jiného příkazu SELECT (poddotaz)
- Názvy sloupců mohou být v tabulkách shodné, proto je vždy lepší názvy sloupců kvalifikovat názvem tabulky „<nazev tabulky>.<nazev sloupce>“
- v rozsáhlých dotazech to může být pracné, proto se obvykle zavádí **aliasy** (korelace) pro názvy tabulek

  … FROM **knihy** [AS] **k** WHERE **k**.nazev LIKE (“*matematika*”)

Možnosti spojení

- **„čárkami oddělené“**
  - vytváří kartézský součin množin záznamů spojovaných tabulek
  - syntakticky nejjednodušší, ale pro spojení tabulek s definovaným vztahem nemá bez omezení shody klíčových v rámci klauzule WHERE velký smysl
  - Alternativou se stejným výsledkem je Křížové spojení, pouze se místo čárek použije „CROSS JOIN“
  - … FROM knihy**,** autori WHERE knihy.autorID=autori.autorID …
  - … FROM knihy **CROSS JOIN** autori WHERE knihy.autorID=autori.autorID …

Možnosti spojení

- **přirozené spojení**
  - spojuje záznamy pro které platí, že se shodují ve všech stejně se jmenujících sloupcích

    … FROM knihy **NATURAL JOIN** autori …
- **spojení s uvedenými sloupci**
  - spojuje záznamy pro které platí, že se shodují ve vyjmenovaných sloupcích

    … FROM knihy **JOIN** autori **USING** (AutorID) …

Možnosti spojení

- **Podmíněné spojení**
  - nejčastější možnost, srovnávací podmínka (ky) je definovaná v klauzuli ON
  - umožňuje, na rozdíl od „přirozeného spojení“ a „spojení s uvedenými sloupci“, srovnávat i sloupce s různými názvy
  - Syntaxe:

    FROM Knihy k {**INNER** | **LEFT [OUTER]** | **RIGHT [OUTER]** | **FULL [OUTER]**} **JOIN** Autori a
    ON k.autorID = a.AutorID …

  - INNER – vnitřní spojení, ve výsledku jsou jen záznamy, ve kterých skutečně dojde ke shodě v porovnávaných polích
  - LEFT, RIGHT a FULL jsou vnější spojení, kdy z jedné (LEFT, RIGHT) nebo obou (FULL) tabulek jsou záznamy všechny, bez ohledu na to, jestli u nich došlo ke shodě v porovnávaných polích

Možnosti spojení

- spojení tabulek pomocí operátoru **UNION**
  - jedná se o jiný druh spojení, než všechna dosud zmiňovaná
  - záznamy nejsou spojovány k sobě (přidávání sloupců), ale za sebe (přidávání řádků)
  - jedná se tedy o operaci sjednocení (duplicitní řádky jsou uvedeny jen jednou) – pokud to není žádoucí, přidává se klíčové slovo ALL
  - podmínkou je aby spojované struktury byly shodné

```
SELECT Jmeno, Prijmeni FROM Tym1

UNION [ALL]

SELECT Jmeno, Prijmeni FROM Tym2
…
```

Další možnosti příkazu SELECT

- přidáním klauzule **INTO** lze zajistit, aby výstup příkazu SELECT byl uložen do tabulky
  - pokud již existuje, bude tímto příkazem přepsána

```
SELECT Jmeno, Prijmeni INTO VybraneOsoby FROM Osoby
WHERE RokNarozeni < 1990;
```

- Přidáním klauzule TOP (LIMIT – MySQL, ROWNUM – Oracle) lze omezit počet výstupních řádků

```
SELECT TOP 3 * FROM Osoby ORDER BY RokNarozeni DESC;
```

Příkaz INSERT INTO

- slouží pro vkládání hodnot do tabulky
- synataxe:

INSERT INTO <tabulka> [(<název sloupce> [{, <název sloupce>} …])]
VALUES (<hodnota> [{, <název sloupce>} …])

- pokud nejsou zadány názvy sloupců, musí být zadány hodnoty pro všechny sloupce ve stejném pořadí jako v tabulce
- pokud jsou sloupce zadány, musí odpovídat pořadí hodnot pořadí zápisu, bez ohledu na to jak je to v tabulce

Příkaz INSERT INTO

- Příklad:

```
INSERT INTO osoby
VALUES ('Eva','Novotná')

INSERT INTO osoby (Jmeno, Prijmeni)
VALUES ('Eva','Novotná')
```

- Klauzule VALUES může být nahrazena příkazem SELECT. V tom případě může být vloženo více záznamů najednou. Výsledek příkazu SELECT musí mít správný počet a pořadí hodnot jako je v tabulce, případně ve výpisu

```
INSERT INTO osoby
SELECT jmeno, prijmeni FROM kurz WHERE kurzID = 10
```

Příkaz UPDATE

- slouží pro aktualizaci dat
- Syntaxe:

  UPDATE <tabulka>

  SET <název sloupce> = <výraz hodnoty> [{, **<název sloupce> =** <hodnota> }…]
  [**WHERE** <podmínka>]

- <výraz hodnoty> samozřejmě může být také konstanta

  UPDATE zbozi

  SET cena = cena * 0.95, stav = ‘akce’
  WHERE mnozstvi > 100

Příkaz UPDATE

- záznamy, které mají být aktualizovány, mohou být určeny také pomocí spojení tabulek nebo pomocí predikátu s IN nebo EXISTS

```
UPDATE zboží INNER JOIN vybrane ON zboží.zboziID = vybrane.ID
SET cena = cena * 0.95, stav = ‘akce’;

UPDATE zboží
SET cena = cena * 0.95, stav = ‘akce’
WHERE zboziID IN (SELECT ID FROM vybrane)

UPDATE zboží
SET cena = cena * (SELECT koeficient FROM akce WHERE akceID = (SELECT Akce ID FROM vybrane)),
stav = ‘akce’
WHERE zboziID IN (SELECT ID FROM vybrane)
```

Příkaz DELETE

- slouží pro mazání dat
- pokud mají být smazány všechny záznamy, lze nahradit příkazem TRUNCATE
- syntaxe:

DELETE FROM <tabulka>
WHERE <podmínka>

- podmínka má stejnou syntaxi jako v příkazu SELECT

DELETE FROM zboží
WHERE zboziID IN (SELECT ID FROM vybrane);

DELETE FROM zboží; je stejné jako TRUNCATE TABLE zboží; (pozor ne DROP TABLE zboží;)

Poddotazy

- vrací jeden řádek (hodnotu)
  - v klauzuli WHERE, porovnání s jednou hodnotou (WHERE pole = (SELECT pole FROM …))
  - v příkazu INSERT INTO pro náhradu hodnoty v klauzuli VALUES
- Vrací více řádků (hodnot)
  - viz klauzule WHERE, porovnání pomocí predikátů s IN, EXISTS (WHERE pole IN (SELECT …)
  - v klauzuli FROM, kde výsledek poddotazu nahrazuje tabulku (… FROM (SELECT …)…)
  - v příkazu INSERT INTO, při vkládání dat z jiného dotazu místo klauzule VALUES
  - v příkazech UPDATE, DELETE v klauzuli WHERE s predikáty IN, EXISTS
- Poddotazy mohou být použity opakovaně (vnořované do sebe)

Zdroje

- Kroenke, David a Auer, David J. Databáze. 1. vyd. Brno: Computer Press, 2015. 496 s. ISBN 978-80-251-4352-0.
- Sheldon, Robert. SQL: začínáme programovat. 1. vyd. Praha: Grada, 2005. 499 s. Průvodce. ISBN 80-247-0999-6.