

Databázové systémy

Úvod do databázových systémů

Vilém Vychodil

KMI/DATA1, Přednáška 1

Databázové systémy

Přednáška 1: Přehled

1 Základní pojmy:

- databázový systém, formální model/implementace,
- model dat, dotazovací jazyk,
- architektura databázového systému,
- systém řízení báze dat.

2 Přehled modelů dat:

- souborový model, síťový model, hierarchický model,
- relační model,
- objektové modely, relačně/objektové modely,
- další modely (modely pro semistrukturovaná data).







3 Přehled relačních systémů řízení báze dat:

- uzavřená/otevřená řešení,
- PostgreSQL (základní charakteristika),
- příklady práce s databází.







Přehled kursu

- 1 Úvod do databázových systémů
- 2 Relační model dat
- 3 Základní operace s relacemi
- 4 Přirozené spojení
- 5 Relační operace odvozené ze spojení
- 6 Sumarizace, vnořené dotazy, dělení
- 7 Integritní omezení
- 8 Úvod do funkčních závislostí
- 9 Reprezentace hierarchických struktur
- 10 Transakční zpracování dat
- 11 Fyzická struktura databáze
- 12 Algoritmy pro vyhodnocování dotazů

Literatura (hlavní zdroje)

-  Date C. J.: *Database in Depth: Relational Theory for Practitioners*
O'Reilly Media 2005, ISBN 978-0596100124
-  Date C. J.: *An Introduction to Database Systems*
Addison-Wesley 2003, ISBN 978-0321197849
-  Date C. J.: *SQL and Relational Theory: How to Write Accurate SQL Code*
O'Reilly Media 2011, ISBN 978-1449316402
-  Date C. J., Darwen H.: *Foundation for Object/Relational Databases*
Addison-Wesley Professional 1998, ISBN 978-0201309782
-  Date C. J., Darwen H.: *Databases, Types and the Relational Model*
Addison Wesley 2006, ISBN 978-0321399427
-  Date C. J.: *Logic and Databases: The Roots of Relational Theory*
Trafford Publishing 2007, ISBN 978-1425122904

Literatura (vedlejší zdroje)

-  Abiteboul S., Hull R., Vianu V.: *Foundations of Databases: The Logical Level* Addison-Wesley 1994, ISBN 978-0201537710
-  Atzeni P., Batini C., De Antonellis V.: *Relational Database Theory* Addison Wesley 1993, ISBN 978-0805302493
-  Celko J.: *Joe Celko's Trees and Hierarchies in SQL for Smarties* Morgan Kaufmann 2012, ISBN 978-0123877338
-  Garcia-Molina H., Ullman J., Widom J.: *Database Systems: The Complete Book* Prentice Hall 2008, ISBN 978-0131873254
-  Maier D: *Theory of Relational Databases* Computer Science Press 1983, ISBN 978-0914894421
-  Simovici D.: Tenney R.: *Relational Database Systems* Academic Press 1995, ISBN 978-0126443752

Co je databázový systém

databáze (angl.: *data base*):

- *kolekce perzistentních dat* používaných aplikacemi nějakého subjektu
- **perzistence** = data přetrvávají výpočetní proces, který je vytvořil
- příklad: subjekt = univerzita, aplikace = studijní agenda, výzkumná agenda

databázový systém, angl.: *database system*

Systém pro organizaci, definici, manipulaci a dotazování nad perzistentními daty, který lze popsat jako množinu algoritmů pracujících s daty v určeném tvaru.

často chápán dvojím způsobem:

- 1 jako teorie, tj. **formální model** (přesně definovaný a který lze zkoumat)
- 2 jako konkrétní **softwarová implementace** vycházející z teorie (viz bod 1)

pro naše účely: **data** = **informace** (nerozlišujeme význam)

Co je model dat

(formální) model dat, angl.: *data model*

Množina abstraktních a soběstačných formálních definic datových struktur a operací s daty (případně dalších operací, omezení a podobně), které dohromady tvoří formální výpočetní model, se kterým mohou uživatelé interagovat.

poznámky:

- existuje několik různých formálních modelů dat (viz přehled dále)
- stále je potřeba rozlišovat formální model × jeho implementace

model dat (určitého subjektu)

Přesněji: **model databáze** – návrh nebo implementace organizace dat určitého subjektu (např. návrh organizace dat „studijní agendy“ subjektu „univerzita“).

Formální model × dotazovací jazyk

kategorie jazyků souvisejících s formálními modely

- **dotazovací jazyk** (angl.: *query language*, zkráceně QL) = jazyk pro vyjadřování dotazů (angl.: *queries*) pro získávání dat z databáze
 - **jazyk pro definici dat** (angl.: *data definition language*, zkráceně DDL) = jazyk pro popis typu a struktury dat, která budou v databázi uložena
 - **jazyk pro modifikaci dat** (angl.: *data modification language*, zkráceně DML) = jazyk pro vkládání, aktualizaci a mazání dat v databázi
-
- jazyky používají (typicky) uživatelé různých rolí:
 - **administrátor databáze** – jazyk pro definici dat
 - **uživatel databáze** – dotazovací jazyk, jazyk pro modifikaci dat
 - k jednomu formálnímu modelu typicky existuje *víc jazyků* dané kategorie (!!!)
(např. pro relační model dat jsou jazyky SQL, QUEL, Tutorial D, ...)

Tři vrstvy architektury databázového systému

fyzická vrstva, angl.: *physical level*:

- nejnižší vrstva, zabývá se fyzickým (efektivním a perzistentním) uložením dat
- zajímavá z pohledu implementace DB systému, pro uživatele (téměř) nezajímavá

logická vrstva, angl.: *logical level*:

- vrstva mezi fyzickou a externí vrstvou, abstrahuje od fyzického uložení dat

externí vrstva, angl.: *external level*:

- definuje, jakým způsobem jsou data reprezentována pro konkrétní uživatele
- umožňuje individuální pohled na databázi (poskytuje individuální služby)

poznámky:

- budeme se zabývat převážně logickou a externí vrstvou, fyzickou minimálně
- ANSI/SPARC Study Group on Database Management Systems (1975)

System řízení báze dat (SŘBD)

system řízení báze dat, angl.: *database management system (DBMS)*

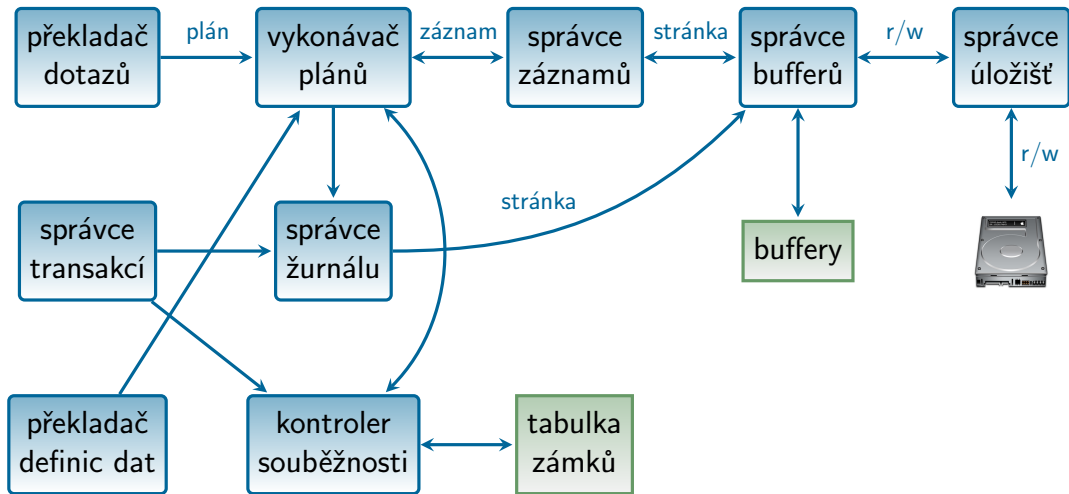
Programový celek implementující databázový systém vycházející z určitého formálního modelu dat a poskytující následující služby:

- souběžný **víceuživatelský přístup** k databázi (neblokované zpracování dotazů),
- **transakční zpracování dat** (atomicita, konzistence, izolace, trvanlivost),
- **perzistentní uložení dat** a systém **zotavení z chyb** (žurnálování dat),
- **integritní omezení** (prevence vytvoření nesmyslných nebo nekonzistentních dat),
- **bezpečnost přístupu k datům** (autorizovaný přístup, šifrování),
-

poznámky:

- složitostí implementace jsou vyspělé SŘBD srovnatelné s operačními systémy
- „malé (účelově vytvořené) SŘBD“ neposkytují všechny uvedené služby

Typická struktura SŘBD



Přehled modelů dat (paradigmat databázových systémů)

- **souborový model** – historicky nejstarší (cca 1955–)
začátky: Grace M. Hopper, jazyk FLOW-MATIC (později COBOL)
- **síťový model** – Charles Bachman (1969, vývoj trval enormní dobu)
grafový pohled na schéma databáze (model je komplikovaný a přežitý)
- **hierarchický model** – IBM (cca 1960, přežitý model ale zažívá renesanci, XML)
lze chápat jako zjednodušení síťového modelu (grafy jsou nahrazeny stromy)
- **relační model** – Edgar F. Codd (1969, rychlý rozmach, dnes mainstream)
model založený na pojmu n -ární relace s úzkou vazbou na predikátovou logiku
- **objektové modely** – mnoho modelů, 1989 Statice (Symbolics Inc.)
perzistentní uložení objektů, obvykle omezené možnosti dotazování
- **relačně/objektové modely** – víc návrhů, různá úroveň, 1990–
pokusy o doplnění objektových rysů do relačního modelu
- **další modely** – vše, co se nevešlo do předchozí klasifikace (hodně)
modely pro key-value databáze, semistrukturovaná data, XML databáze, ...

Přehled paradigmat: Souborový model

charakteristika:

- historicky první databázové systémy, dávkové zpracování dat (50. léta)
- data uložena jako množina záznamů stejného typu v souborech (*angl.: flat files*)
- textový/binární formát souborů (např. řádky v CSV souborech/bloky oktetů)
- omezené využití (soubory `/etc/passwd`, `/etc/groups`, `/etc/shadow`, ...)

vlastnosti:

- ⊕ jednoduchý systém lze snadno udělat „na koleně“ (absence SŘBD)
- ⊖ *nedostatečná abstrakce* (prakticky se jedná o model fyzické vrstvy)
- ⊖ *omezené možnosti dotazování* (jako „vrať záznam na dané pozici v souboru“)
- ⊖ nechtěná *redundance dat* (jedna data uložena na víc místech)
- ⊖ *neexistence transakcí* (možnost uvést databázi do nekonzistentního stavu)
- ⊖ komplikované nebo *nemožné sdílení* mezi síťovými aplikacemi

Přehled paradigmat: Síťový model

charakteristika:

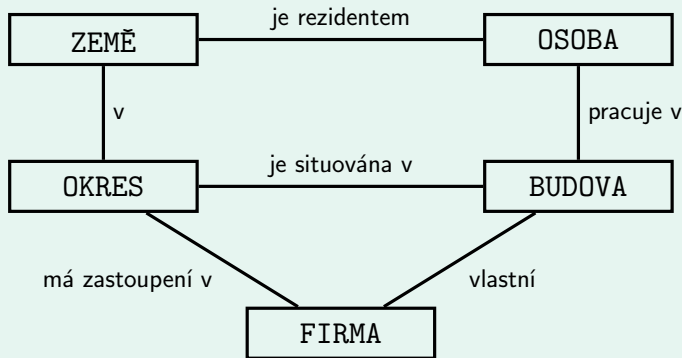
- překonané paradigma (stále používané, především na systémech typu mainframe),
- databáze jsou organizovány pomocí dvou základních typů databázových objektů:
 - **záznamy** (angl.: *records*) – obsahují pojmenované položky (datové jednotky)
 - **odkazy** (angl.: *links*) – reprezentují vazby mezi záznamy (= ukazatele)
- IDMS (1973–), model ožívá jako „grafové databáze“ (např. Neo4j)

vlastnosti:

- ⊕ vykonávání dotazů může být extrémně rychlé (při dobrém návrhu databáze)
- ⊕ reprezentace dat může být úsporná (při dobrém návrhu databáze)
- ⊖ formální model je extrémně složitý (komplikovaná analýza modelu)
- ⊖ dotazování je málo deklarativní, převládá procedurální manipulace s ukazateli
- ⊖ *asymetrie v dotazech* (komplikované nebo nemožné *ad hoc* dotazy)

Příklad (Síťový diagram v síťovém modelu)

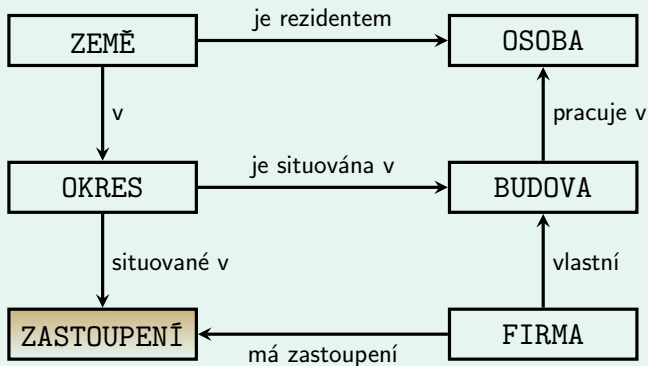
- **typy záznamů** (ne jednotlivé záznamy) jsou zakresleny jako uzly
- **odkazy** jsou zakresleny jako pojmenované neorientované hrany



poznámka: odkazy mohou být typu 1 : 1, 1 : N, nebo M : N

Příklad (Diagram datové struktury v síťovém modelu)

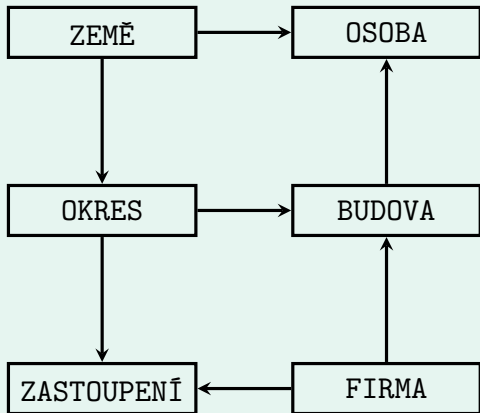
- odkazy jsou pouze typu 1 : 1 nebo 1 : N a zakreslují se jako orientované hrany
- odkazy typu $M : N$ se redukují na odkazy 1 : N přidáním pomocného záznamu
- orientované hrany lze chápat jako *ukazatel* (jdeme „proti směru šipky“)



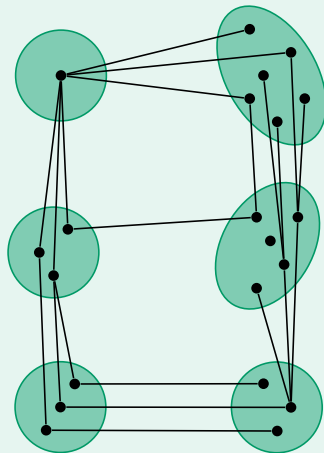
asymetrie: „V které budově pracuje osoba?“ × „Které osoby pracují v budově?“

Příklad (Fyzická datová struktura v síťovém modelu)

- obecně existuje víc záznamů daného typu (záznamy značeny ●)



→



Přehled paradigmat: Hierarchický model

charakteristika:

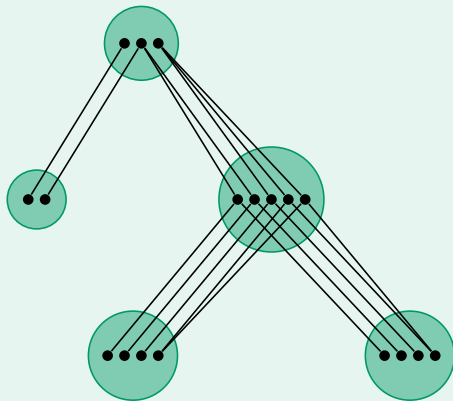
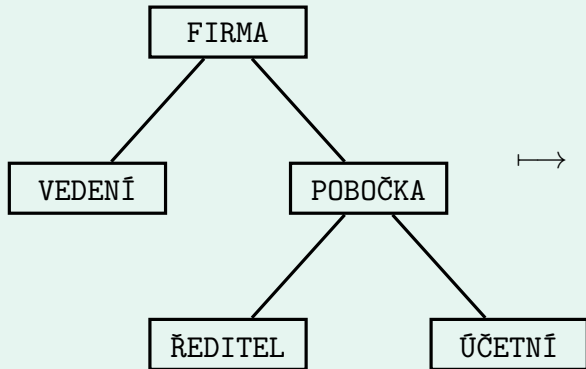
- zjednodušený pohled na síťový model – diagram struktury je (uspořádaný) strom:
 - **záznamy** (angl.: *records*) – obsahují pojmenované položky (datové jednotky)
 - **odkazy** (angl.: *links*) – reprezentují vazby mezi záznamy (= ukazatele)
- přitom musí být splněny následující podmínky:
 - každý typ záznamu (až na kořenový) má právě jednoho **předchůdce**
 - je definováno **pořadí potomků** všech uzlů
- implementačně jednodušší než obecné síťové modely, IMS (IBM, 1966–)

vlastnosti:

- ⊕ vykonávání dotazů může být extrémně rychlé (při dobrém návrhu databáze)
- ⊕ reprezentace dat může být úsporná (při dobrém návrhu databáze)
- ⊖ *dotazování* je založeno na *prohledávání stromů* (do hloubky/šířky)
- ⊖ asymetrie v dotazech

Příklad (Diagram struktury v hierarchickém modelu)

- typy záznamů a odkazy = *strom*
- záznamy a fyzické ukazatele = *množina stromů*



Přehled paradigmat: Relační model

charakteristika:

- jeden typ databázových objektů: **relace (nad relačními schématy)** je
 - matematický pojem n -ární relace = formální protějšek pojmu „datová tabulka“
 - formalizuje **základní data**, **výsledky dotazů** i **vztahy mezi daty**
- Codd, E. F.: A relational model of data for large shared data banks
Communications of the ACM 13: 6 (1970)

vlastnosti:

- ⊕ dobrý teoretický model, který lze navíc efektivně implementovat
- ⊕ od počátku formalizuje i související fenomény (závislosti v datech, normalizace)
- ⊕ **logická nezávislost dat** – fyzická a logická vrstva je oddělena
- ⊕ model je **referenčně transparentní**
- ⊕ k dispozici hodně kvalitativně různých SRBD cílených na různou klientelu
- ⊖ čistý relační model žádný (komerčně nasaditelný) SRBD neimplementuje

Příklad („Datové tabulky“ v relačním modelu dat)

jmeno	id	rodne-cislo
Adams	12345	571224/4023
Black	33355	840525/6670
Chang	66066	891117/1024

stuID	rok	predmet	typ
12345	2013	KMI/DATA1	A
12345	2013	KMI/FJ	B
33355	2012	KMI/DATA1	A
33355	2013	KMI/DATA2	B
33355	2013	KMI/PP1	A
66066	2012	KMI/DATA1	C

stuID	rok	predmet	vysl	datum
12345	2013	KMI/DATA1	95%	18/01/13
12345	2013	KMI/FJ	FAIL	25/06/13
12345	2013	KMI/FJ	35%	27/06/13
33355	2012	KMI/DATA1	FAIL	18/01/13
66066	2012	KMI/DATA1	FAIL	19/01/13
66066	2012	KMI/DATA1	85%	06/02/13

Přehled paradigmat: Objektové databáze

charakteristika:

- objektová databáze = perzistentní objektový systém
- Statice (1989, první komerčně použitelný systém), Elephant (open-source řešení)

vlastnosti:

- ⊕ odpadá mapování databázových elementů na objekty v programovacím jazyku
- ⊕ práce s objekty je principiálně stejná jako s neperzistentními objekty
- ⊕ podporuje očekávané objektové rysy, např. dědičnost
- ⊖ neexistuje rozumný (a jednoduchý) formální model
- ⊖ serializace může být výkonnostní problém (u komplikovaných struktur)
- ⊖ v mnoha ohledech je podobné síťovému modelu (vzájemné odkazy mezi objekty)
- ⊖ dotazování je málo deklarativní, převládá procedurální manipulace
- ⊖ podporuje pouze jednoduché typy dotazů (typicky vyhledávání podle rovnosti)

Příklad (Perzistentní objektová databáze elephant)

```
(ql:quickload "elephant")
```

```
(use-package :elephant)
```

```
(defclass pair ())
```

```
  ((x :accessor pair-x :initarg :x :index t)
```

```
   (y :accessor pair-y :initarg :y))
```

```
  (:metaclass persistent-metaclass))
```

```
(with-open-store (*connection-spec*))
```

```
  (let* ((a (make-instance 'pair :x 100 :y nil))
```

```
         (b (make-instance 'pair :x 200 :y a)))
```

```
    (setf (pair-y a) b)))
```

```
(with-open-store (*connection-spec*))
```

```
  (get-instances-by-value 'pair 'x 200))
```

Přehled paradigmat: Objektové × relační databáze

charakteristika:

- snaha *zkombinovat* relační model a objektové paradigma
- mnoho koncepčně různých (správných i nesprávných) přístupů

dva hlavní typy přístupů:

- 1 **nesprávný přístup:** perverzní *rozšíření relačního modelu o reference*
 - ➖ relační model *de facto* degeneruje na síťový model
 - ➖ všechny výhody relačního modelu jsou ztraceny
- 2 **správný přístup:** přijmeme fakt, že *relační typy = třídy* a *hodnoty = objekty*
 - ➕ zavádí subtypování do relačního modelu (koerce obecně ne)
 - ➕ je referenčně transparentní, objekty lze pouze konstruovat, ne mutovat
 - ➕ zachovává flexibilitu relačního dotazování (pokud je model dobře implementovaný)



Date C. J., Darwen H.: *Foundation for Object/Relational Databases*
Addison-Wesley Professional 1998, ISBN 978-0201309782

Přehled paradigmat: Další modely

key-value databáze

- perzistentní asociační struktura (ukládání/vyhledávání hodnot podle klíčů)
- Berkeley DB (C, Java verze), Redis (RAM databáze, možnost perzistence)
- ⊕ rychlost, spolehlivost, velká míra nasazení
- ⊖ obvykle pouze jednoduché typy dotazování

modely pro semistrukturovaná data

- semistrukturovaná data je poněkud „vágní pojem“
- dokumentově orientované databáze (XML, YAML, JSON, BSON, . . .)
- MongoDB, CouchDB, OrientDB
- výhodami a nevýhodami *podobné síťovému modelu*

poznámka:

- obskurní pojem „NoSQL“ označující nerelační databáze (!!!)

Dostupné relační SŘBD

uzavřená řešení: Oracle (Oracle), MS SQL Server (Microsoft), DB2 (IBM), ...

- ⊕ obvykle dobrá podpora, stabilita, ověřeno dlouhým provozem
- ⊖ cena, software je blackbox (často obří monolit)

otevřená řešení:

- Ingres (1973, UC Berkeley, <http://www.actian.com/products/ingres>)
 - komerční podpora od Actian Corporation; podporuje SQL a QUEL
- MariaDB (1995, komunitní fork MySQL, <http://mariadb.org/>)
 - velký počet nasazení, nezávislé storage engines (fyzická vrstva DB)
- PostgreSQL (1985, <http://postgresql.org/>)
 - vyzrálý „velký“ databázový systém, dobrá programovatelnost
- SQLite (2000, <http://sqlite.org/>)
 - embedded databáze, nepotřebuje spuštěný server, nejpoužívanější SQL engine

PostgreSQL

historie vývoje:

- UC Berkeley (1986, M. Stonebraker) – projekt navazující na databázi Ingres
- nejprve dotazovací jazyk POSTQUEL, později SQL (Postgres95, PostgreSQL)
- současnost (9. září 2013): verze 9.3

důležité rysy:

- ⊕ drží se standardů: implementuje ISO SQL
- ⊕ „velká databáze“ funkčně a výkonově srovnatelná s komerčními produkty
- ⊕ stabilita a spolehlivost
- ⊕ programovatelnost (PL/pgSQL, PL/Perl, PL/Python)
- ⊕ rozšiřitelnost (možnost doprogramovat SŘBD podle potřeb)
- ⊕ detailní dokumentace (<http://postgresql.org/docs/manuals/>)
- ⊕ platformová nezávislost

Příklad (PostgreSQL, použití interaktivního klienta psql)

```
$ psql -h slon.inf.upol.cz slondb -U vychodil ENTER
```

```
psql (9.1.9)
```

```
slondb=> \connect mojedb ENTER
```

```
mojedb=> \dt ENTER
```

```
⋮
```

```
mojedb=> SELECT * FROM katedra WHERE zkratka = 'KI'; ENTER
```

id	jmeno	zkratka
1024	katedra informatiky	KI

(1 row)

```
mojedb=> \quit ENTER
```

Příklad (PostgreSQL, příklad použití v PHP5)

```
$db = pg_connect ("host=slon.inf.upol.cz_dbname=mojedb" .  
                 "user=vychodil_password=heslo");  
  
$query = "SELECT_id,_jmeno_FROM_katedra_WHERE_zkratka=_'KI'";  
  
$result = pg_query ($query);  
  
while ($tuple = pg_fetch_array ($result, NULL, PGSQL_ASSOC)) {  
    printf ("ID:_%s,_JMENO:_%s\n",  
           $tuple ["id"], $tuple ["jmeno"]);  
}  
  
pg_free_result ($result);  
  
pg_close ($db);
```

Příklad (PostgreSQL, použití v Common LISPu, balík postmodern)

```
(ql:quickload "postmodern")  
  
(use-package :postmodern)  
  
(with-connection '("mojedb" "vychodil" "heslo" "slon.inf.upol.cz")  
  (doquery (:select 'id 'jmeno  
                :from 'katedra  
                :where (:= 'zkratka "KI"))  
    (id jmeno)  
    (format t "ID:~A,~AJMENO:~A~%" id jmeno)))
```

poznámky:

- <http://marijnhaberbeke.nl/postmodern/> (instalovatelné přes quicklisp)
- odstraňuje „prkenné“ psaní dotazů ve formě řetězců (makra generující dotazy)

Přednáška 1: Závěr

pojmy k zapamatování:

- databáze, model dat, systém řízení báze dat,
- dotazovací jazyk, jazyk pro definici/modifikaci dat,
- fyzická/logická/externí vrstva databázového systému,
- přehled modelů: souborový, síťový, hierarchický, relační, ostatní.

použité zdroje:



Date C. J.: *Database in Depth: Relational Theory for Practitioners*
O'Reilly Media 2005, ISBN 978-0596100124



Garcia-Molina H., Ullman J., Widom J.: *Database Systems: The Complete Book*
Prentice Hall 2008, ISBN 978-0131873254



Tsichritzis D., Lochovsky F.: *Data Base Management Systems*
Academic Press 1976, ISBN 978-0127017402