**BI-SPOL-9 Relační databáze, dotazování v relační algebře, základní koncepce jazyka SQL (SELECT, DDL, DML, DCL, TCL), vyjádření integritních omezení v DDL**

BI-DBS

Základem relační databáze jsou relace, což jsou vlastně databázové tabulky. Ty se skládají ze sloupců (atributů) a z řádků (záznamů). Relace je celá tabulka.

Jednotlivé atributy pak mohou plnit například funkci primárního klíče, cizího klíče atd.

### Relační databáze

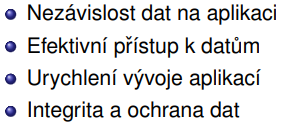
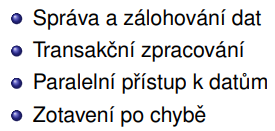
Databáze je **soubor záznamů** (zpráv), jako jsou znaky, čísla, diagramy, jejichž **systematická struktura** umožňuje, aby tyto zprávy mohly být vyhledávány pomocí počítače.

Zabývá se řízením velkého množství, perzistentních, spolehlivých a sdílených dat

* **velkého množství** – pro data nestačí operační paměť
* **perzistentních** – data přetrvávají od zpracování ke zpracování
* **spolehlivých** – data lze rekonstruovat po chybě
* **sdílených** – data jsou přístupná více uživatelům (užívání na základě přístupových práv, koordinované současné využívání stejných dat více uživateli)
* Existence dat v DB je nezávislá na aplikačních programech.
* DBS = databázový systém, DBS = DBMS + DB
* DB = databáze
* **DBMS** = systém řízení bází dat (database managment system)
* RDBMS (relační) – implementuje relační model dat

**Smysl a přínos databázové technologie**

* Potřeba oddělit data od aplikací, zapouzdřit je a jednotně k nim přistupovat – vznik databázových technologií
* Hlavní přínosy databázové technologie:



**Rozhraní RDBMS**

Komunikace s databázovým strojem probíhá pomocí dotazovacího jazyka SQL

* DDL – definice dat (Data Definition Language)
* DML – úprava dat a dotazování (Data Manipulation Language)
* TCL – řízení transakcí (Transaction Control Language)
* DCL – přístupová práva (Data Control Language)

**Schéma relační databáze**

**Obsah obrázku text

Popis byl vytvořen automaticky**

**Obsah obrázku text

Popis byl vytvořen automaticky**

Relace je množina prvků, z nichž každý se tváří jako řádek tabulky. Relační databáze je potom dána množinou relací a množinou integritních omezení.

**Relace** – množina n-tic ⊂ D**1** × D**2**... × D***n*** (relace = tabulka)

* Jména atributů [A1, A2, …, An]
* Domény atributů Di – název, typ a rozsah
* n-tice (a1, a2, …, an) je prvek relace
* jméno relace R
* schéma relace R(A1 : D1, A2 : D2, …, An : Dn), zkráceně R(A)

nepřesně: (relace = tabulka), (schéma relace = záhlaví tabulky)

**Obsah obrázku text

Popis byl vytvořen automaticky**

**Obsah obrázku text

Popis byl vytvořen automaticky**

**Příklad relace**

****

* Pro práci s relacemi můžeme použít jazyk SQL, relační algebru nebo relační kalkul

**Obsah obrázku text

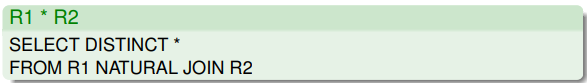
Popis byl vytvořen automaticky**

### Relační algebra

* je vyšší jazyk, nespecifikuje „jak se mají věci udělat“, ale „co má být výsledkem“
* **výsledkem** dotazu **je** **relace**, která může být vstupem do dalšího dotazu – dotazy lze řetězit
* řeší pouze dotazování nikoliv DML a DDL
* velmi specializovaný jazyk
* jazyk, který umožňuje realizovat RA je relačně úplný (např SQL)
* výrazy (dotazy) se skládají z operací a operandů
* operandem je vždy celá relace

#### Základní operace relační algebry

* **selekce** (restrikce) relace R dle podmínky φ (logický výraz).
  + R(φ) = def {u | u ∈ R ∧ φ(u)},  *(množina splnujiči podminku φ)*
  + Výběr řádku tabulky dle určitého kritéria
* **projekce** relace R na množinu atributů C, kde C ⊆ A
  + R[C] = def {u[C] | u ∈ R}. *(vyběr attributu)*
  + Výběr sloupců tabulky dle názvů
* **přirozené spojení** relací R(A) a S(B) s výsledkem T(C)
  + R∗S = def {u | u[A] ∈ R ∧ u[B] ∈ S), kde C = A u B a kde výběr n-tic pro spojení je dán rovností na všech průnikových atributech A a B *(výbeř n-tic  ⇔ rovností na všech prunikových atributech A a B.)*
  + Spojení 2 relací dle rovnosti všech společných atributů (např. pokud obě relace obsahují PK atribut – R2[id], R3[id], tak výsledná relace sjednotí řádky se stejným id)

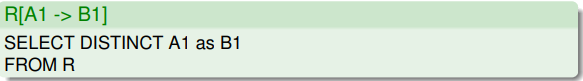


* **obecné spojení** (Θ-spojení) R(A) a S(B) s výsledkem T(C)
  + R[t1 Θ t2]S =def {u | u[A] ∈ R, u[B] ∈ S, u.t1Θu.t2}, kde Θ ∈ {<, >, =, ≤, ≥, !=} nebo logická spojka {∧, ∨, ¬} pro konstrukci složitějších podmínek spojení. C je „zřetězením“ A a B – spol. atributy ve výsledku se zopakují, nutno je přejmenovat nebo používat kvalifikaci, např. R.A.
  + Spojení dvou relací dle poskytnuté rovnosti/nerovnosti vybraných atributů

Obsah obrázku text

Popis byl vytvořen automaticky

* **přejmenování** atributu
  + t → alias.



* **množinové** operace

Obsah obrázku text

Popis byl vytvořen automaticky

**Obsah obrázku text

Popis byl vytvořen automaticky**

Obsah obrázku text

Popis byl vytvořen automaticky

**Vyhodnocování výrazu v RA**

* vyhodnocuje se zleva doprava
* operace selekce a projekce mají přednost před binárními operacemi
* pro změnu priority používáme složené závorky

#### Obecné polospojení

Obsah obrázku text

Popis byl vytvořen automaticky

#### Přirozené polospojení

Obsah obrázku text

Popis byl vytvořen automaticky

* Polospojení (obecné i přirozené) lze chápat jako „syntaktickou zkratku“:
  + Spojení následované projekcí na A resp. B
* Polospojení není to samé co LEFT/RIGHT JOIN
* Skutečná implementace je obvykle efektivnější než spojení a selekce



#### Antijoin

Podmnožina n-tic z R, jenž nejsou spojitelné s žádnou n-ticí z S (doplněk JOIN)

**Obsah obrázku text

Popis byl vytvořen automaticky**

#### Relační dělení

**Obsah obrázku text

Popis byl vytvořen automaticky**

#### Příklady RA

Obsah obrázku text

Popis byl vytvořen automatickyObsah obrázku text

Popis byl vytvořen automaticky

Obsah obrázku text

Popis byl vytvořen automaticky

Obsah obrázku text

Popis byl vytvořen automaticky

Obsah obrázku text

Popis byl vytvořen automaticky

### Základní koncepce jazyka SQL

**SQL**

* dotazovací jazyk pro práci s relačními DB - součástí je DDL, DML, TCL, DCL
* říkáme co chceme získat, ne jak se to má dělat
* intuitivní zápis, připomíná anglické věty
* syntaxe úzce spojená s RA

**DDL** (Data Definition Language)

* jazyk pro definici dat, pro definici integritních omezení
* CREATE, ALTER, DROP TABLE, …
* *CREATE TABLE teachers (id int, name varchar(255));*

**DML** (Data Manipulation Language)

* jazyk pro manipulaci dat a dotazování
* *SELECT, INSERT, UPDATE*, DELETE, …

**TCL** (Transaction Control Language)

* jazyk pro řízení transakcí
* COMMIT, ROLLBACK

**DCL** (Data Control Language)

* jazyk pro přiřazení přístupových práv
* GRANT <příkaz> ON <table> TO <user>

#### SELECT

* Pro výběr dat (vyhledávání), dotazování na data
* zobrazuje tabulku na základě vybraných atributů a podmínek

Obsah obrázku text

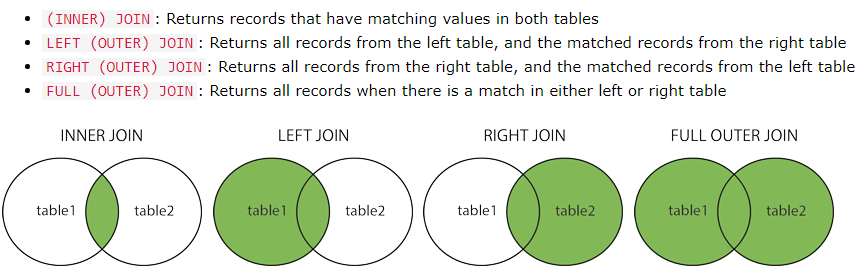
Popis byl vytvořen automaticky

Obsah obrázku text

Popis byl vytvořen automaticky

Obsah obrázku text

Popis byl vytvořen automaticky

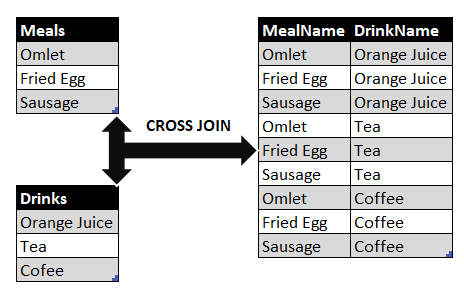


**NATURAL JOIN** – spojí tabulky přes stejný název sloupce – ten zobrazí jen jednou (INNER JOIN by zobrazil dva stejné sloupce)

Obsah obrázku text

Popis byl vytvořen automaticky

**CROSS JOIN** – generuje páry každé řádky z tabulky1 s každou řádkou tabulky2



**SELF JOIN** – klasický JOIN jen s tabulka se sebou samou

Obsah obrázku text

Popis byl vytvořen automaticky

Obsah obrázku text

Popis byl vytvořen automaticky

**COALESCE** – hodnoty NULL jsou nahrazeny definovanou hodnotou

Obsah obrázku text

Popis byl vytvořen automaticky

**DISTINCT** – zabrání výpisu duplicit, vztahuje se na celou SELECT klauzuli

Obsah obrázku text, křížovka

Popis byl vytvořen automaticky

**ORDER BY** – specifikuje řazení



**UNION** – zkombinuje výsledky dvou a více SELECTů

* všechny musí mít stejný počet sloupců, stejné data types, stejně seřazené
* SELECT City FROM C UNION SELECT City FROM B ORDER BY City
  + Zobrazí seznam měst

**Alias**

Obsah obrázku text

Popis byl vytvořen automaticky

**Operátory porovnání**

**Obsah obrázku stůl

Popis byl vytvořen automaticky**

* v dotazu můžeme porovnávat vůči NULL – IS NULL, IS NOT NULL

**Agregace**

Obsah obrázku text

Popis byl vytvořen automaticky



Obsah obrázku text

Popis byl vytvořen automaticky

Obsah obrázku text

Popis byl vytvořen automaticky

Obsah obrázku text

Popis byl vytvořen automaticky

**GROUP BY**

* určíme, podle čeho (jakého sloupce) máme seskupovat řádky dohromady
* např. když chceme zobrazit počet herců podle filmu, kde v tabulka vypadá jako na obrázku. Musíme použít group by, protože to jinak SQL nedokáže napárovat
* Obsah obrázku stůl

  Popis byl vytvořen automaticky

**HAVING**

* Podmínka pro seskupení – např, že chceme seskupit filmy, které mají počet herců větší než jedna

Obsah obrázku text

Popis byl vytvořen automaticky

**Pořadí vyhodnocení**

Obsah obrázku text

Popis byl vytvořen automaticky

**Nevztažené poddotazy**

* SELECT můžeme „vnořovat“

Obsah obrázku text

Popis byl vytvořen automaticky

Obsah obrázku text

Popis byl vytvořen automaticky

Obsah obrázku text

Popis byl vytvořen automaticky

Obsah obrázku text

Popis byl vytvořen automaticky**CASE**

* Jako if-then-else
* Obsah obrázku text

  Popis byl vytvořen automaticky
* Obsah obrázku text

  Popis byl vytvořen automaticky
* Obsah obrázku text

  Popis byl vytvořen automaticky

**LIKE**

Obsah obrázku text

Popis byl vytvořen automaticky

**Množinový predikát IN**

**Obsah obrázku text

Popis byl vytvořen automaticky**

**Obsah obrázku text

Popis byl vytvořen automaticky**

**ANY** – vrací TRUE, jestli některá z hodnot odpovídá podmínce, používáme se subquery

**ALL** – vrací TRUE, jestli všechny hodnoty odpovídají podmínce, používáme se subquery

**UNIQUE** – vrací TRUE, když všechny vrácené řádky jsou unikátní, používáme se subquery

**EXISTS**

* Existenční kvantifikátor, TRUE pokud vrátí alespoň 1 řádek

Obsah obrázku text

Popis byl vytvořen automaticky

Všeobecný kvantifikátor můžeme udělat pomocí negE (negP(x))

Množinové operace

**UNION** – sjednocení (nebo)

**INTERSECT** – průnik (a zároveň)

**EXCEPT/MINUS** – rozdíl (a nejsou)

**DDL** (Data Definition Language)

* **CREATE** – CREATE DATABASE databasename  
  *Obsah obrázku text

  Popis byl vytvořen automaticky*
* **DROP** – DROP DATABASE databasename  
  **
* **ALTER**  
  *Obsah obrázku text, oranžová

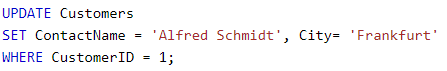
  Popis byl vytvořen automaticky*

**DML** (Data Manipulation Language)

* **SELECT**
* **INSERT  
  **

****

* **UPDATE**  
  Obsah obrázku text

  Popis byl vytvořen automaticky 
* **DELETE  
  **
* ****

**TCL** (Transaction Control Language)

Session – jedno navázané spojení s DB serverem. V session probíhají transakce. Lze mít otevřeno více session i pod jedním DB uživatelem.

* **COMMIT** – potvrzení změn – všechny provedené změny jsou perzistentně uloženy v databázi, ostatní session od této chvíle provedené změny vidí
* **ROLLBACK** – odvolání změn – všechny změny provedené v příslušné transakci jsou odvolány
* **SAVEPOINT** – možnost definování „značky“ uvnitř transakce, ke které lze vztáhnout ROLLBACK
* Můžeme mít zapnutý autocommit
  + ON – každý DML příkaz je automaticky potvrzen
  + OFF – musí přijít explicitní COMMIT nebo ROLLBACK

**DCL** (Data Control Language)

Právo na: SELECT, INSERT, UPDATE, DELETE, ALTER, EXECUTE, INDEX, REFERENCE

* GRANT – přidá právo
* REVOKE – odebere právo

Obsah obrázku text

Popis byl vytvořen automaticky

### Integritní omezení v DDL

* způsob pro zavedení pravidel pro určité sloupce, tabulky…

**Okamžik kontroly**

* dají se dočasně vypnout (např. během transakce odstraňování/přidávání dat)
* nastavit čas kontroly
* odložitelné na konec transakce
* zpětné zapnutí IO může/nemusí vyžadovat kontrolu platnosti dat již vložených do DB

#### Deklarativní

* hlídá DBMS

**IO sloupce**

* NOT NULL
* DEFAULT
* UNIQUE
* PRIMARY KEY
* REFERENCES
* CHECK

**IO tabulky**

* stejné jako pro sloupce

#### Procedurální

* kontrola i na straně klienta
* např. TRIGGER

#### Příklad

Obsah obrázku text

Popis byl vytvořen automaticky