

## Kapitola 3: Relační model

- Struktura relačních databází
- Relační algebra
- $n$ -ticový relační kalkul
- Doménový relační kalkul
- Rozšířené operace relační algebry
- Modifikace databáze
- Pohledy

### Základní struktura

- Mějme množiny  $A_1, A_2, \dots, A_n$  *relace*  $r$  je podmnožina kartézského součinu  $A_1 \times A_2 \times \dots \times A_n$ . Tedy *relace*  $r$  je *množina  $n$ -tic*  $(a_1, a_2, \dots, a_n)$ , kde  $a_i \in A_i$
- Příklad: je-li  
 $jméno = \{Novák, Starý, Coufal, Liška\}$   
 $ulice = \{Hlavní, Severní, Sadová\}$   
 $město = \{Hradec, Rájec, Polná\}$   
 pak  $r = \{(Novák, Hlavní, Harrison), (Starý, Severní, Rájec), (Coufal, Severní, Rájec), (Liška, Sadová, Polná)\}$  je *relace* na  $jméno \times ulice \times město$

### Relační schéma

- $A_1, A_2, \dots, A_n$  jsou *atributy*
- $R = (A_1, A_2, \dots, A_n)$  je *relační schéma*  
 $zákazník-schéma = (jméno, ulice, město)$
- $r(R)$  je *relace* (pojmenování) na relačním schématu  $R$   
 $zákazník$  (*zákazník-schéma*)  
 $zákazník$  (*jméno, ulice, město*)

### Instance relace

- Současné hodnoty (*instance relace*) jsou specifikovány tabulkou.
- Element  $t$  *relace*  $r$  je  $n$ -*tice* reprezentovaná *řádkem* v tabulce.

<i>jméno</i>	<i>Ulice</i>	<i>město</i>
Novák	Hlavní	Hradec
Starý	Severní	Rájec
Coufal	Severní	Rájec
Liška	Sadová	Polná

*zákazník*

## Klíče

- Necht'  $K \subseteq R$
- $K$  je *superklíč* schématu  $R$ , jestliže hodnoty  $K$  jsou dostatečné pro jednoznačnou identifikaci  $n$ -tice každé možné relace  $r(R)$ .  
Např.:  $\{\text{jméno}, \text{ulice}\}$  a  $\{\text{jméno}\}$  jsou superklíče schématu *zákazník*, jestliže dva zákazníci nemají shodné jméno.
- $K$  je *kandidátní klíč*, jestliže  $K$  je minimální.  
Např.:  $\{\text{jméno}\}$  je kandidátní klíč schématu *zákazník*, jestliže dva zákazníci nemají shodné jméno a žádná jeho podmnožina není superklíč.

## Odvozování klíčů z E-R množin

- **Silná entitní množina.** Primární klíč množiny se stává primárním klíčem relace.
- **Slabá entitní množina.** Primární klíč relace je sjednocení primárního klíče silné množiny entit a parciálního klíče slabé množiny.
- **Množina vztahů.** Sjednocení primárních klíčů spojených množin entit se stává superklíčem relace. Pro binární vztahy mnoho na mnoho je superklíč též primární klíč. Pro binární vztahy mnoho na jednu se stává primární klíč množiny entit „mnoho“ primárním klíčem relace. Pro vztahy jedna na jednu může být primární klíč relace z každé množiny entit.

## Dotazovací jazyky

- Jazyk, kterým uživatel žádá informace z databáze.
- Kategorie jazyků:
  - *Procedurální* = uživatel specifikuje jaké operace má systém provést pro získání výsledku; uživatel určí posloupnost kroků k výpočtu výsledku
  - *Neprocedurální* = uživatel určí jak má vypadat výsledek, ale neříká jak výsledek vytvořit
- Čisté jazyky („*pure*“):
  - Relační algebra (procedurální)
  - $N$ -tiový relační kalkul (neprocedurální)
  - Doménový relační kalkul (neprocedurální)
- Čisté jazyky jsou stručné, formální, bez jakéhokoli syntaktického cukru.
- Čisté jazyky jsou základem dotazovacích jazyků, které se běžně používají.

## Relační algebra

- Procedurální jazyk
- 6 základních operátorů
  - výběr (selekce)
  - projekce
  - sjednocení
  - rozdíl množin
  - kartézský součin
  - přejmenování
- Operátory berou jednu nebo více relací jako vstup a vrací novou relaci jako výsledek.

### Operace výběr (selekce)

- Značení:  $\sigma_P(r)$
- Definováno jako:

$$\sigma_P(r) = \{t \mid t \in r \text{ and } P(t)\}$$

Kde  $P$  je výraz v propozičním kalkulu, skládající se z podmínek ve formě:

<atribut> = <atribut> nebo <konstanta>

≠

>

≥

<

≤

„spojené pomocí“:  $\wedge$  (**and**),  $\vee$  (**or**),  $\neg$  (**not**)

### Příklad selekce

- Relace  $r$ :

$A$	$B$	$C$	$D$
$\alpha$	$\alpha$	1	7
$\alpha$	$\beta$	5	7
$\beta$	$\beta$	12	3
$\beta$	$\beta$	23	10

- $\sigma_{A=B \wedge D > 5}(r)$  – vybere řádky, kde  $A = B$  a zároveň  $D > 5$

$A$	$B$	$C$	$D$
$\alpha$	$\alpha$	1	7
$\beta$	$\beta$	23	10

## Operace projekce

- Značení:  $\Pi_{A_1, A_2, \dots, A_k}(r)$   
kde  $A_1, A_2, \dots, A_k$  jsou jména atributů a  $r$  je jméno relace.
- Výsledek je definován jako relace  $k$  sloupců, které dostaneme smazáním sloupců, které nejsou vyjmenovány
- Duplicitní řádky jsou z výsledku odstraněny, protože relace je množina.

## Příklad projekce

- Relace  $r$ :

A	B	C
$\alpha$	10	1
$\alpha$	20	1
$\beta$	30	1
$\beta$	40	2

- $\Pi_{A, C}(r)$

A	C
$\alpha$	1
<del><math>\alpha</math></del>	1
$\beta$	1
$\beta$	2

=

A	C
$\alpha$	1
$\beta$	1
$\beta$	2

## Operace sjednocení

- Značení:  $r \cup s$
- Definováno jako:
 
$$r \cup s = \{t \mid t \in r \text{ or } t \in s\}$$
- Aby bylo sjednocení  $r \cup s$  platné,
  - $r$  a  $s$  musí mít *stejnou aritu* (stejný počet atributů)
  - Domény atributů musí být kompatibilní (např. 2. sloupec  $r$  obsahuje hodnoty stejného typu jako 2. sloupec  $s$ )

## Příklad sjednocení

- Relace  $r, s$ :

A	B
$\alpha$	1
$\alpha$	2
$\beta$	1

$r$

A	B
$\alpha$	2
$\beta$	3

$s$

- $r \cup s$

A	B
$\alpha$	1
$\alpha$	2
$\beta$	1
$\beta$	3

## Operace rozdíl množin

- Značení:  $r - s$
- Definováno jako:
 
$$r - s = \{t \mid t \in r \text{ and } t \notin s\}$$
- Rozdíly množin musí být brány mezi kompatibilními relacemi.
  - $r$  a  $s$  musí mít stejnou aritu
  - domény atributů relací  $r$  a  $s$  musí být kompatibilní

### Příklad rozdílu množin

- Relace  $r, s$ :

$A$	$B$
$\alpha$	1
$\alpha$	2
$\beta$	1

$r$

$A$	$B$
$\alpha$	2
$\beta$	3

$s$

- $r - s$

$A$	$B$
$\alpha$	1
$\beta$	1

## Operace kartézského součinu

- Značení:  $r \times s$
- Definováno jako:
 
$$r \times s = \{t \ q \mid t \in r \text{ and } q \in s\}$$
- Předpokládá se, že atributy relací  $r(R)$  a  $s(S)$  jsou disjunktní (tj.  $R \cap S = \emptyset$ ).
- Pokud atributy relací  $r(R)$  a  $s(S)$  nejsou disjunktní, musí se přejmenovat.

### Příklad kartézského součinu

- Relace  $r, s$ :

A	B
$\alpha$	1
$\beta$	2

$r$

C	D	E
$\alpha$	10	+
$\beta$	10	+
$\beta$	20	-
$\gamma$	10	-

$s$

- $r \times s$

A	B	C	D	E
$\alpha$	1	$\alpha$	10	+
$\alpha$	1	$\beta$	10	+
$\alpha$	1	$\beta$	20	-
$\alpha$	1	$\gamma$	10	-
$\beta$	2	$\alpha$	10	+
$\beta$	2	$\beta$	10	+
$\beta$	2	$\beta$	20	-
$\beta$	2	$\gamma$	10	-

### Skládání operací

- Můžeme tvořit výrazy skládáním operací
- Příklad:  $\sigma_{A=C}(r \times s)$
- Operace *přirozené spojení* (natural join):
  - Značení:  $r \bowtie s$
  - Nechť  $r$  a  $s$  jsou relace na schématech  $R$  a  $S$ . Výsledek je relace na schématu  $R \cup S$ , kterou získáme uvažováním každého páru  $n$ -tic  $t_r$  z  $r$  a  $t_s$  z  $s$ .
  - Mají-li  $t_r$  a  $t_s$  stejné hodnoty na každém atributu z  $R \cap S$ , je k výsledku přidána  $n$ -tice  $t$  taková, že
    - \*  $t$  má na atributech z  $R$  stejnou hodnotu jako  $t_r$
    - \*  $t$  má na atributech z  $S$  stejnou hodnotu jako  $t_s$

Příklad:

$R = (A, B, C, D)$

$S = (E, B, D)$

- Výsledné schéma =  $(A, B, C, D, E)$
- $r \bowtie s$  je definováno jako:

$$r \bowtie s = \Pi_{r.A, r.B, r.C, r.D, s.E} (\sigma_{r.B=s.B \wedge r.D=s.D} (r \times s))$$

## Příklad přirozeného spojení (Natural join operation)

- Relace  $r, s$ :

A	B	C	D
$\alpha$	1	$\alpha$	a
$\beta$	2	$\gamma$	a
$\gamma$	4	$\beta$	b
$\alpha$	1	$\gamma$	a
$\delta$	2	$\beta$	b

 $r$ 

B	D	E
1	a	$\alpha$
3	a	$\beta$
1	a	$\gamma$
2	b	$\delta$
3	b	$\varepsilon$

 $s$ 

- $r \bowtie s$

A	B	C	D	E
$\alpha$	1	$\alpha$	a	$\alpha$
$\alpha$	1	$\alpha$	a	$\gamma$
$\alpha$	1	$\gamma$	a	$\alpha$
$\alpha$	1	$\gamma$	a	$\gamma$
$\delta$	2	$\beta$	b	$\delta$

## Operace dělení

- Značení:  $r \div s$
- Odpovídá dotazům, které obsahují frázi „pro všechny“.
- Necht'  $r$  a  $s$  jsou relace na schématech  $R$  a  $S$ , kde
  - $R = (A_1, \dots, A_m, B_1, \dots, B_n)$
  - $S = (B_1, \dots, B_n)$
 Výsledek operace  $r \div s$  je relace na schématu  $R - S = (A_1, \dots, A_m)$ 

$$r \div s = \{t \mid t \in \Pi_{R-S}(r) \wedge \forall u \in s: tu \in r\}$$
- Vlastnost
  - Necht'  $q = r \div s$
  - Pak  $q$  je největší relace, pro kterou platí:  $q \times s \subseteq r$
- Definice pomocí podmínek základních operací algebry. Necht'  $r(R)$  a  $s(S)$  jsou relace a necht'  $S \subseteq R$ 

$$r \div s = \Pi_{R-S}(r) - \Pi_{R-S}((\Pi_{R-S}(r) \times s) - \Pi_{R-S,S}(r))$$
- Vysvětlení:
  - $\Pi_{R-S,S}(r)$  jednoduše přeřadí atributy  $r$
  - $\Pi_{R-S}((\Pi_{R-S}(r) \times s) - \Pi_{R-S,S}(r))$  dává ty  $n$ -tice  $t$  v  $\Pi_{R-S}(r)$ , že pro nějakou  $n$ -tici  $u \in s$ ,  $tu \notin r$

## Příklad dělení

- Relace  $r, s$ :

$A$	$B$
$\alpha$	1
$\alpha$	2
$\alpha$	3
$\beta$	1
$\gamma$	1
$\delta$	1
$\delta$	3
$\delta$	4
$\delta$	6
$\varepsilon$	1
$\varepsilon$	2

 $r$ 

$B$
1
2

 $s$ 

- $r \div s$

$A$
$\alpha$
$\varepsilon$

## Jiný příklad dělení

- Relace  $r, s$ :

$A$	$B$	$C$	$D$	$E$
$\alpha$	a	$\alpha$	a	1
$\alpha$	a	$\gamma$	a	1
$\alpha$	a	$\gamma$	b	1
$\beta$	a	$\gamma$	a	1
$\beta$	a	$\gamma$	b	3
$\gamma$	a	$\gamma$	a	1
$\gamma$	a	$\gamma$	b	1
$\gamma$	a	$\beta$	b	1

 $r$ 

$D$	$E$
a	1
b	1

 $s$ 

- $r \div s$

$A$	$B$	$C$
$\alpha$	a	$\gamma$
$\gamma$	a	$\gamma$

## Operace přiřazení

- Operace přiřazení ( $\leftarrow$ ) představuje vhodný způsob, jak vyjádřit komplexní dotazy; dotazy se píšou jako sekvenční program skládající se ze skupiny přiřazení následovaných výrazem, jehož hodnota je zobrazena jako výsledek dotazu.
- Přiřazení musí být vždy prováděno na dočasné relační proměnné.
- Příklad: Napišme  $r \div s$  jako
 
$$\begin{aligned} temp1 &\leftarrow \Pi_{R-S}(r) \\ temp2 &\leftarrow \Pi_{R-S}((temp1 \times s) - \Pi_{R-S,S}(r)) \\ result &= temp1 - temp2 \end{aligned}$$
  - Výsledek vpravo od  $\leftarrow$  je přiřazen do relační proměnné vlevo od  $\leftarrow$ .

## Příklady dotazů

- Najděte všechny zákazníky, kteří mají (zároveň) účet aspoň v pobočkách „Praha-Vršovice“ a „Praha-Spořilov“.
  - Dotaz 1
 
$$\Pi_{JZAK}(\sigma_{JPOB = \text{„Praha-Vršovice“}}(vkladatel \bowtie \acute{u}\acute{c}et)) \cap \Pi_{JZAK}(\sigma_{JPOB = \text{„Praha-Spořilov“}}(vkladatel \bowtie \acute{u}\acute{c}et))$$

kde *JZAK* značí *jméno-zákazníka*, *JPOB* značí *jméno-pobočky* a *MPOB* značí *město-pobočky*
  - Dotaz 2
 
$$\Pi_{JZAK, JPOB}(vkladatel \bowtie \acute{u}\acute{c}et) \div \rho_{temp(JPOB)}(\{\text{„Praha-Vršovice“}, \text{„Praha-Spořilov“}\})$$
- Najděte všechny zákazníky, kteří mají účet ve všech pobočkách v Praze.
 
$$\Pi_{JZAK, JPOB}(vkladatel \bowtie \acute{u}\acute{c}et) \div \Pi_{JPOB}(\sigma_{MPOB = \text{„Praha“}}(pobočka))$$

## n-ticový relační kalkul (Tuple relational calculus)

- Neprocedurální dotazovací jazyk, kde každý dotaz je ve tvaru
 
$$\{t \mid P(t)\}$$
- Je to množina všech n-tic  $t$  takových, že predikát  $P$  je pravdivý pro  $t$
- $t$  je n-ticová proměnná (*tuple variable*);  $t[A]$  značí hodnotu n-tice  $t$  na atributu  $A$
- $t \in r$  značí, že n-tice  $t$  je v relaci  $r$
- $P$  je výraz (*formula*) podobný výrazu v predikátové logice (predikátovém kalkulu)

## Výraz v predikátovém kalkulu

- Množina atributů a konstant
- Množina operátorů porovnání: (např.:  $<$ ,  $\leq$ ,  $=$ ,  $\neq$ ,  $>$ ,  $\geq$ )
- Množina spojek: and ( $\wedge$ ), or ( $\vee$ ), not ( $\neg$ )
- Implikace ( $\Rightarrow$ ):  $x \Rightarrow y$ , jestliže  $x$  je pravdivé, pak  $y$  také
 
$$x \Rightarrow y \equiv \neg x \vee y$$
- Množina kvantifikátorů:
  - $\exists t \in r(Q(t)) \equiv$  existuje n-tice  $t$  z relace  $r$  taková, že platí  $Q(t)$
  - $\forall t \in r(Q(t)) \equiv$  pro všechny n-tice  $t$  v relaci  $r$  platí  $Q(t)$

## Příklad (banka)

*pobočka* (*pobočka-jméno*, *pobočka-město*, *aktivum*)  
*zákazník* (*zákazník-jméno*, *zákazník-ulice*, *zákazník-město*)  
*účet* (*pobočka-jméno*, *číslo-úctu*, *částka*)  
*půjčka* (*pobočka-jméno*, *půjčka-číslo*, *částka*)  
*vkladatel* (*zákazník-jméno*, *číslo-úctu*)  
*dlužník* (*zákazník-jméno*, *půjčka-číslo*)

## Příklady dotazů

- Najděte *pobočka-jméno*, *půjčka-číslo* a *částka* pro půjčky přes 1200

$$\{t \mid t \in \text{půjčka} \wedge t[\text{částka}] > 1200\}$$

- Najděte číslo půjčky pro každou půjčku s částkou větší než 1200

$$\{t \mid \exists s \in \text{půjčka} (t[\text{půjčka-číslo}] = s[\text{půjčka-číslo}] \wedge s[\text{částka}] > 1200)\}$$

Poznámka:  $t$  je relace na schématu (*půjčka-číslo*) protože to je jediný zmíněný atribut  $n$ -tice  $t$ .

- Najděte jména všech zákazníků, kteří mají v bance půjčku, účet nebo obojí

$$\{t \mid \exists s \in \text{dlužník}(t[\text{zákazník-jméno}] = s[\text{zákazník-jméno}]) \vee$$

$$\exists u \in \text{vkladatel}(t[\text{zákazník-jméno}] = u[\text{zákazník-jméno}])\}$$

- Najděte jména všech zákazníků, kteří mají v bance půjčku a účet

$$\{t \mid \exists s \in \text{dlužník}(t[\text{zákazník-jméno}] = s[\text{zákazník-jméno}]) \wedge$$

$$\exists u \in \text{vkladatel}(t[\text{zákazník-jméno}] = u[\text{zákazník-jméno}])\}$$

- Najděte jména všech zákazníků, kteří mají půjčku v pobočce Praha-Spořilov

$$\{t \mid \exists s \in \text{dlužník}(t[\text{zákazník-jméno}] = s[\text{zákazník-jméno}] \wedge$$

$$\exists u \in \text{půjčka}(u[\text{pobočka-jméno}] = \text{„Praha-Spořilov“} \wedge u[\text{půjčka-číslo}] = s[\text{půjčka-číslo}])\}$$

- Najděte jména všech zákazníků, kteří mají půjčku v pobočce Praha-Spořilov, ale nemají účet v žádné pobočce banky.

$$\{t \mid \exists s \in \text{dlužník}(t[\text{zákazník-jméno}] = s[\text{zákazník-jméno}] \wedge$$

$$\exists u \in \text{půjčka}(u[\text{pobočka-jméno}] = \text{„Praha-Spořilov“} \wedge u[\text{půjčka-číslo}] = s[\text{půjčka-číslo}])$$

$\wedge$

$$\neg \exists v \in \text{vkladatel} (v[\text{zákazník-jméno}] = t[\text{zákazník-jméno}])\}$$

- Najděte jména všech zákazníků, kteří mají půjčku v pobočce Praha-Spořilov a města, ze kterých jsou

$$\{t \mid \exists s \in \text{půjčka} (s[\text{pobočka-jméno}] = \text{„Praha-Spořilov“} \wedge$$

$$\exists u \in \text{dlužník} (u[\text{půjčka-číslo}] = s[\text{půjčka-číslo}] \wedge$$

$$t[\text{zákazník-jméno}] = u[\text{zákazník-jméno}] \wedge$$

$$\exists v \in \text{zákazník} (u[\text{zákazník-jméno}] = v[\text{zákazník-jméno}] \wedge$$

$$t[\text{zákazník-město}] = v[\text{zákazník-město}]))))\}$$

- Najděte jména všech zákazníků, kteří mají účet ve všech pobočkách umístěných v Praze:

$$\{t \mid \forall s \in \text{pobočka} (s[\text{pobočka-město}] = \text{„Praha“} \Rightarrow$$

$$\exists u \in \text{účet} (s[\text{pobočka-jméno}] = u[\text{pobočka-jméno}] \wedge$$

$$\exists s \in \text{vkladatel} (t[\text{zákazník-jméno}] = s[\text{zákazník-jméno}] \wedge$$

$$s[\text{číslo-úctu}] = u[\text{číslo-úctu}]))))\}$$

### Bezpečnost výrazů

- V kalkulu n-tic je možné psát výrazy, které generují nekonečné relace
- Např.  $\{t \mid \neg(t \in r)\}$  vede k nekonečné relaci, je-li doména aspoň jednoho atributu relace  $r$  nekonečná
- Abychom tomuto zamezili, omezíme množinu použitelných výrazů na *bezpečné* výrazy
- Výraz  $\{t \mid P(t)\}$  v relačním kalkulu n-tic je *bezpečný*, jestliže se každá komponenta  $t$  objeví v  $dom(P)$ .
- $dom(P)$  je doména výrazu  $P$  – hodnoty konstant a n-tic vyskytujících se v  $P$  a také všechny hodnoty relací jejichž názvy se vyskytují v  $P$ .

### Doménový relační kalkul

- Neprocedurální dotazovací jazyk, který je, co se týče síly, ekvivalentní n-ticovému relačnímu kalkulu.
- Každý dotaz je výraz ve tvaru:

$$\{ \langle x_1, x_2, \dots, x_n \rangle \mid P(x_1, x_2, \dots, x_n) \}$$

- $x_1, x_2, \dots, x_n$  reprezentují doménové proměnné
- $P$  reprezentuje formuli podobnou formuli predikátového kalkulu

### Příklady dotazů

- Najděte *pobočka-jméno (b)*, *půjčka-číslo (l)* a *částka (a)* pro půjčky přes 1200:

$$\{ \langle b, l, a \rangle \mid \langle b, l, a \rangle \in \text{půjčka} \wedge a > 1200 \}$$

- Najděte jména všech zákazníků (*c*), kteří mají půjčku přes 1200:

$$\{ \langle c \rangle \mid \exists b, l, a (\langle c, l \rangle \in \text{dlužník} \wedge \langle b, l, a \rangle \in \text{půjčka} \wedge a > 1200) \}$$

- Najděte jména všech zákazníků, kteří mají půjčku u pobočky Praha-Spořilov a částku půjčky:

$$\{ \langle c, a \rangle \mid \exists l (\langle c, l \rangle \in \text{dlužník} \wedge \exists b (\langle b, l, a \rangle \in \text{půjčka} \wedge b = \text{„Praha-Spořilov“})) \}$$

- Najděte jména všech zákazníků, kteří mají půjčku, účet nebo obojí u pobočky Praha-Spořilov:

$$\{ \langle c \rangle \mid \exists l (\langle c, l \rangle \in \text{dlužník} \wedge \exists b, a (\langle b, l, a \rangle \in \text{půjčka} \wedge b = \text{„Praha-Spořilov“})) \}$$

$$\vee \exists a (\langle c, a \rangle \in \text{dlužník} \wedge \exists b, n (\langle b, a, n \rangle \in \text{účet} \wedge b = \text{„Praha-Spořilov“})) \}$$

- Najděte jména všech zákazníků, kteří mají účet ve všech pobočkách umístěných v Praze:

$$\{ \langle c \rangle \mid \forall x, y, z ((\langle x, y, z \rangle \in \text{pobočka} \wedge y = \text{„Praha“}) \Rightarrow$$

$$\exists a, b (\langle x, a, b \rangle \in \text{účet} \wedge \langle c, a \rangle \in \text{vkladatel})) \}$$

### Bezpečnost výrazů

$$\{ \langle x_1, x_2, \dots, x_n \rangle \mid P(x_1, x_2, \dots, x_n) \}$$

je bezpečný výraz, platí-li následující:

- Všechny hodnoty, které se objeví ve všech *n*-ticích celého výrazu, jsou hodnoty z  $\text{dom}(P)$ .
- Pro každou podformuli „existuje“ tvaru  $\exists x (P_1(x))$  platí, že tato podformule je pravdivá jen tehdy, je-li hodnota  $x$  v  $\text{dom}(P_1)$ , takže je  $P_1(x)$  pravdivá.
- Pro každou podformuli „pro všechny“ tvaru  $\forall x (P_1(x))$  platí, že tato podformule je pravdivá jen tehdy, je-li  $P_1(x)$  pravdivá pro všechny hodnoty  $x$  z  $\text{dom}(P_1)$ .

### Rozšířené operace relační algebry

- Zobecněná projekce
- Vnější spojení (Outer join)
- Souhrnné funkce

## Zobecněná projekce

- Rozšiřuje operaci projekce tak, že umožňuje použití aritmetických funkcí v seznamu projekce.

$$\Pi_{F_1, F_2, \dots, F_n}(E)$$

- $E$  je jakýkoliv výraz relační algebry
- Každá z  $F_1, F_2, \dots, F_n$  jsou aritmetické výrazy zahrnující konstanty a atributy ve schématu  $E$ .
- Daná relace *úvěr-info* (*zákazník\_jméno*, *limit*, *stav*) najde, kolik může každá osoba utratit:

$$\Pi_{\text{zákazník\_jméno, limit} - \text{stav}}(\text{úvěr-info})$$

## Vnější spojení (Outer join)

- Rozšíření operace přirozené spojení, které zabraňuje ztrátě informací.
- Spočítá operaci spojení a přidá n-tice z jedné relace, které neodpovídají n-ticím v druhé relaci k výsledkům operace spojení.
- Používá hodnotu *null*:
  - null* značí, že hodnota je neznámá nebo neexistuje
  - všechna porovnávání zahrnující *null* mají z definice hodnotu **false**.

## Příklad

- Relace *půjčka*

<i>pobočka-jméno</i>	<i>půjčka-číslo</i>	<i>částka</i>
Děčín	L-170	3000
Rájec	L-230	4000
Praha-Spořilov	L-260	1700

- Relace *dlužník*

<i>zákazník-jméno</i>	<i>půjčka-číslo</i>
Novák	L-170
Starý	L-230
Hájek	L-155

- $půjčka \bowtie dlužník$

<i>pobočka-jméno</i>	<i>půjčka-číslo</i>	<i>částka</i>	<i>zákazník-jméno</i>
Děčín	L-170	3000	Novák
Rájec	L-230	4000	Starý

- $půjčka \bowtie_{\text{null}} dlužník$

<i>pobočka-jméno</i>	<i>půjčka-číslo</i>	<i>částka</i>	<i>zákazník-jméno</i>
Děčín	L-170	3000	Novák
Rájec	L-230	4000	Starý
Praha-Spořilov	L-260	1700	<i>null</i>

- $půjčka \bowtie_{\text{null}} dlužník$

<i>pobočka-jméno</i>	<i>půjčka-číslo</i>	<i>částka</i>	<i>zákazník-jméno</i>
Děčín	L-170	3000	Novák
Rájec	L-230	4000	Starý
<i>null</i>	L-155	<i>null</i>	Hájek

- *půjčka*  $\bowtie$  *dlužník*

<i>pobočka-jméno</i>	<i>půjčka-číslo</i>	<i>částka</i>	<i>zákazník-jméno</i>
Děčín	L-170	3000	Novák
Rájec	L-230	4000	Starý
Praha-Spořilov	L-260	1700	<i>null</i>
<i>null</i>	L-155	<i>null</i>	Hájek

### Souhrnné funkce

- Souhrnný operátor  $\mathcal{G}$  vezme kolekci hodnot a vrátí jednoduchou hodnotu jako výsledek.

**avg**: průměrná hodnota  
**min**: minimální hodnota  
**max**: maximální hodnota  
**sum**: součet hodnot  
**count**: počet hodnot

$$G_1, G_2, \dots, G_n \mathcal{G}_{F_1 A_1, F_2 A_2, \dots, F_m A_m}(E)$$

- $E$  je jakýkoliv výraz relační algebry
- $G_1, G_2, \dots, G_n$  je seznam atributů podle kterých se  $n$ -tice seskupují
- $F_i$  je souhrnná funkce
- $A_i$  je jméno atributu
- Výsledkem je relace složená z atributů:
  - všechny použité  $G_i$  atributy
  - jeden atribut pro každou použitou souhrnnou funkci

### Příklad

- Relace  $r$

$A$	$B$	$C$
$\alpha$	$\alpha$	7
$\alpha$	$\beta$	7
$\beta$	$\beta$	3
$\beta$	$\beta$	10

- $sum_C(r)$ : 27
- Relace *účet* seskupená podle *pobočka-jméno*:

<i>Pobočka-jméno</i>	<i>číslo-účtu</i>	<i>zůstatek</i>
Praha-Spořilov	A-102	400
Praha-Spořilov	A-201	900
Brno	A-217	750
Brno	A-215	750
Rájec	A-222	700

- *pobočka-jméno*  $\mathcal{G}_{sum\ zůstatek}$  (*účet*)

<i>pobočka-jméno</i>	<i>suma-zůstatek</i>
Praha-Spořilov	1300
Brno	1500
Rájec	700

## Modifikace databáze

- Obsah databáze lze měnit následujícími operacemi:
  - Mazání
  - Vložení
  - Aktualizace
- Tyto operace jsou vyjadřovány operátorem přiřazení.

### Mazání

- Požadavek na mazání je vyjádřen podobně jako dotaz, ale n-tice nejsou poslány uživateli, ale smazány z databáze.
- Lze mazat pouze celé n-tice, nikoli pouze hodnoty ve vybraných atributech.
- V relační algebře je mazání vyjádřeno takto:

$$r \leftarrow r - E$$

kde  $r$  je relace a  $E$  je dotaz relační algebry.

### Příklady mazání

- Smaže všechny záznamy o účtech v pobočce Praha-Spořilov.  

$$\text{účet} \leftarrow \text{účet} - \sigma_{\text{pobočka-jméno} = \text{„Praha-Spořilov“}}(\text{účet})$$
- Smaže všechny záznamy o půjčkách s částkou v rozmezí 0–50.  

$$\text{půjčka} \leftarrow \text{půjčka} - \sigma_{\text{částka} \geq 0 \text{ and } \text{částka} \leq 50}(\text{půjčka})$$
- Smaže všechny účty a vkladatele v pobočkách umístěných v Brně.

$$r_1 \leftarrow \sigma_{\text{pobočka-město} = \text{„Brno“}}(\text{účet} \bowtie \text{pobočka})$$

$$r_2 \leftarrow \Pi_{\text{pobočka-jméno}, \text{číslo-úctu}, \text{zůstatek}}(r_1)$$

$$r_3 \leftarrow \Pi_{\text{zákazník-jméno}, \text{číslo-úctu}}(r_2 \bowtie \text{vkladatel})$$

$$\text{účet} \leftarrow \text{účet} - r_2$$

$$\text{vkladatel} \leftarrow \text{vkladatel} - r_3$$

### Vložení

- Abychom přidali data do relace, buď:
  - specifikujeme n-tici, která má být přidána
  - napíšeme dotaz, jehož výsledek je množina n-tic, která má být přidána
- V relační algebře je vložení vyjádřeno takto:

$$r \leftarrow r \cup E$$

kde  $r$  je relace a  $E$  je dotaz relační algebry.

- Vložení jednoduché n-tice je vyjádřeno tak, že  $E$  může být konstantní relace obsahující jednu n-tici.

## Příklady vložení

- Vložte informaci do databáze: Novák má 1200 na účtu A-973 v pobočce Praha-Spořilov.

$$\text{účet} \leftarrow \text{účet} \cup \{ („Praha-Spořilov“, A-973, 1200) \}$$

$$\text{vkladatel} \leftarrow \text{vkladatel} \cup \{ („Novák“, A-973) \}$$

- Jako dárek poskytněte všem zákazníkům půjček v pobočce Praha-Spořilov účet se zůstatkem 200. Nechť číslo půjčky slouží jako číslo účtu pro nový účet.

$$r_1 \leftarrow (\sigma_{\text{pobočka-jméno} = „Praha-Spořilov“} (\text{dlužník} \bowtie \text{půjčka}))$$

$$\text{účet} \leftarrow \text{účet} \cup \Pi_{\text{pobočka-jméno}, \text{půjčka-číslo}, 200} (r_1)$$

$$\text{vkladatel} \leftarrow \text{vkladatel} \cup \Pi_{\text{zákazník-jméno}, \text{půjčka-číslo}} (r_1)$$

## Aktualizace

- Mechanismus jak změnit hodnoty v n-tici, aniž by musely být změněny *všechny*
- K tomu se používá operátor zobecněné projekce

$$r \leftarrow \Pi_{F_1, F_2, \dots, F_n} (r)$$

–  $F_i$  je  $i$ -tý atribut relace  $r$ , jestliže  $i$ -tý atribut není aktualizován  
nebo

–  $F_i$  je výraz obsahující pouze konstanty a atributy  $r$ , který dává novou hodnotu atributu, jestliže má být  $i$ -tý atribut aktualizován

## Příklady aktualizace

- Zvyšte všechny zůstatky o 5 %.

$$\text{účet} \leftarrow \Pi_{JPOB, \text{ČÚ}, \text{ZůST} \leftarrow \text{ZůST} * 1.05} (\text{účet})$$

kde  $ZůST$ ,  $JPOB$  a  $ČÚ$  jsou *zůstatek*, *pobočka-jméno* a *číslo-účtu*.

- Vyplatte všechny účty se zůstatkem přes \$10,000 6% úrokem, ostatní 5% úrokem.

$$\text{účet} \leftarrow \Pi_{JPOB, \text{ČÚ}, \text{ZůST} \leftarrow \text{ZůST} * 1.06} (\sigma_{\text{BAL} > 10000} (\text{účet})) \cup$$

$$\Pi_{JPOB, \text{ČÚ}, \text{ZůST} \leftarrow \text{ZůST} * 1.05} (\sigma_{\text{BAL} \leq 10000} (\text{účet}))$$

## Pohledy

- V některých případech není vhodné, aby všichni uživatelé viděli celý logický model databáze (tj. všechny relace aktuálně uložené v databázi).
- Představme si osobu, která potřebuje vědět o počtu půjček zákazníkům, ale nemusí vidět částku půjčky. Tato osoba by měla vidět relaci popsanou v relační algebře takto:

$$\Pi_{\text{zákazník-jméno}, \text{půjčka-číslo}} (\text{dlužník} \bowtie \text{půjčka})$$

- Jakákoli relace, která není součástí koncepčního modelu, ale je viditelná uživateli jako „virtuální relace“, se nazývá *pohled*.

## Definice pohledu

- Pohled je definován příkazem **create view**, který má tvar  

$$\text{create view } v \text{ as } \langle \text{výraz dotazu} \rangle$$
kde  $\langle \text{výraz dotazu} \rangle$  je jakýkoliv správný výraz relační algebry. Jméno pohledu je reprezentováno proměnnou  $v$ .
- Jakmile je pohled definován, jeho jméno může být použito pro odkazování na virtuální relaci, která pohled generuje.
- Definice pohledu není to samé jako vytvoření nové relace vyhodnocením dotazovacího výrazu. Definice pohledu způsobuje uložení výrazu, aby mohl být substituován do dotazů, které používají tento pohled.

## Příklady pohledů

- Představme si pohled (pojmenovaný *všichni-zákazníci*) skládající se z poboček a jejich zákazníků.

**create view všichni-zákazníci as**

$$\Pi_{\text{pobočka-jméno, zákazník-jméno}} (\text{vkladatel} \bowtie \text{účet}) \\ \cup \Pi_{\text{pobočka-jméno, zákazník-jméno}} (\text{dlužník} \bowtie \text{účet})$$

- Nyní můžeme najít všechny zákazníky pobočky Praha-Spořilov napsáním:

$$\Pi_{\text{zákazník-jméno}} (\sigma_{\text{pobočka-jméno} = \text{„Praha-Spořilov“}} (\text{všichni-zákazníci}))$$

## Aktualizace přes pohledy

- Modifikace databáze s použitím pohledů musí být překládány na modifikace aktuálních relací v databázi.
- Představme si uživatele, který potřebuje vidět všechna data o půjčkách v relaci *půjčka* s výjimkou *částky*. Pohled, který ji poskytneme (*pobočka-půjčka*), je definován takto:
 

**create view pobočka-půjčka as**  $\Pi_{\text{pobočka-jméno, půjčka-číslo}} (\text{půjčka})$

 Umožníme-li, aby se jméno pohledu objevilo kdekoliv, kde je akceptováno jméno relace, uživatel může psát:
 
$$\text{pobočka-půjčka} \leftarrow \text{pobočka-půjčka} \cup \{(\text{„Praha-Spořilov“}, \text{L-37})\}$$
- Předchozí vkládání musí být reprezentováno vkládáním do aktuální relace *půjčka*, ze které byl pohled *pobočka-půjčka* vytvořen.
- Vkládání do *půjčka* vyžaduje hodnotu pro *částka*. Vkládání se může chovat takto:
  - odmítne vkládání a vrátí chybové hlášení
  - nebo
  - vloží n-tici („Praha-Spořilov“, L-37, *null*) do relace *půjčka*

## Pohledy definované použitím jiných pohledů

- V definování pohledu je možné použít jen jeden jiný pohled.
- Relace pohledu  $v_1$  závisí přímo na relaci pohledu  $v_2$ , jestliže  $v_2$  je použit ve výrazu, který definuje  $v_1$ .
- Relace pohledu  $v_1$  závisí na relaci pohledu  $v_2$ , jestliže v grafu závislostí je cesta z  $v_2$  do  $v_1$ .
- Relace pohledu  $v$  je *rekurzivní*, jestliže závisí sama na sobě.

## Expanze pohledu

- Cesta, jak definovat význam pohledů definovaných podmínkami ostatních pohledů.
- Necht' pohled  $v_1$  je definován výrazem  $e_1$ , který může sám obsahovat použití relací pohledů.
- Expanze pohledu výrazu opakuje následující nahrazovací krok:
  - repeat**
    - najdi jakoukoli relaci  $v_i$  v  $e_1$
    - nahraď relaci pohledu  $v_i$  výrazem definujícím  $v_i$
  - until** v  $e_1$  nejsou přítomny další relace pohledů
- Pokud definice pohledů nebudou rekurzivní, tato smyčka skončí.