http://www.dcs.fmph.uniba.sk/~plachetk

/TEACHING/DB1

Tomáš Plachetka

Fakulta matematiky, fyziky a informatiky,
Univerzita Komenského, Bratislava

Zima 2021–2022

Formalizácia dotazov

- **Databáza je štruktúra** pre relačný (predikátový) kalkul
- **Dotaz je formula** $\varphi(X_1, ..., X_n)$, kde $X_1, ..., X_n$ sú voľné premenné
- **Výsledok dotazu** $\varphi(X_1, ..., X_n)$ je množina usporiadaných n-tíc $[X_1,...,X_n]$, pre ktoré platí $\varphi(X_1,...,X_n)$

Ako **počítať** výsledok dotazu? O tom je **relačná algebra**. Operandami relačnej algebry sú relácie. Relácia sa dá reprezentovať tabuľkou, kde mená atribútov označujú stĺpce tabuľky (hlavičkový riadok). Zatiaľ predpokladajme, že tabuľka je **množina** riadkov (bez duplikátov)

V ďalšom texte bude **X** označovať vektor atribútov $[X_1, ..., X_n]$, **Y** bude označovať vektor atribútov $[Y_1, ..., Y_n]$ a podobne

Význam relačnej algebry

- **Formálna špecifikácia SQL.** Hoci syntax SELECT je veľmi „barokná“, sémantika sa dá presne popísať relačnou algebrou
- **Optimalizácia dotazov** v DBMS. SELECT sa dá priamočiaro transformovať na ekvivalentný výraz relačnej algebry (operátorový strom)
  - Pri optimalizácii sa tento výraz transformuje na ekvivalentný, výpočtovo efektívnejší algebraický výraz (resp. na postupnosť priradení), ktorému sa hovorí **logický plán**
  - Logický plán zapísaný v relačnej algebre sa namapuje do **fyzických operátorov** konkrétneho DBMS. Takto vznikne **fyzický plán** výpočtu dotazu

Význam relačnej algebry

Pohľad programátora:

• Dotaz, resp. program **v relačnom kalkule**, **Datalogu** a **SQL** vyjadruje **ČO** treba vypočítať (**nie ako**). Nie je príliš dôležité, ako "efektívne" je program napísaný. O optimalizáciu sa stará stroj, nie programátor

• Dotaz, resp. program **v relačnej algebre** vyjadruje **AKO** sa vypočíta výsledná relácia z EDB relácií

(V konečnom dôsledku z návodu AKO sa niečo počíta vyplýva aj ČO sa počíta. Lenže ľudský spôsob myslenia je „najskôr ČO, až potom AKO", t.j. „najskôr špecifikácia, až potom implementácia")

Základné operátory relačnej algebry

- **Zjednotenie, prienik, rozdiel** (vyžaduje sa, aby relácie mali rovnakú schému, t.j. aby boli rovnakého typu)
- **Selekcia**: výber riadkov
- **Projekcia**: výber stĺpcov
- **Kartézsky súčin a join**: skladanie relácií
- **Premenovanie** relácií a atribútov relácií
- ...

Zjednotenie, prienik, rozdiel

Relačná algebra | Relačný kalkul:

$r_1 \cup r_2 =$ $\{\mathbf{X}: r_1(\mathbf{X}) \vee r_2(\mathbf{X})\}$

$r_1 \cap r_2 =$ $\{\mathbf{X}: r_1(\mathbf{X}) \wedge r_2(\mathbf{X})\}$

$r_1 - r_2 =$ $\{\mathbf{X}: r_1(\mathbf{X}) \wedge \neg r_2(\mathbf{X})\}$

Všimnite si, že **negácia sa vyjadruje ako rozdiel relácií** (inak sa ani vyjadriť nedá). Relácia r_1 vystupuje ako „pozitívny kontext“, t.j. obsahuje množinu kandidátov na výsledok. Negácia spôsobí len vynechanie niektorých n-tíc z r_1 (tých, ktoré sú v r_2)

Projekcia: π

Kalkul:

Nech r je typu r(**X**, **Y**). Potom

$\pi_X (r) = \{\mathbf{X}: \exists \mathbf{Y}\ r(\mathbf{X}, \mathbf{Y})\}$

Procedurálna definícia: $r_2 := \pi_{\mathbf{X}} (r_1)$

- r_2 vzniká kopírovaním riadkov r_1, pričom pre každý riadok r_1 sa do r_2 skopírujú len tie atribúty, ktoré sú v **X**
- Nakoniec treba eliminovať duplikované riadky v r_2 (ak počítame s množinami)

Projekcia: π

Príklad (Ullman)

Relation sells

Bar	Beer	Price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Miller	3.00

prices := $\pi_{\text{Beer, Price}}$(sells)

Beer	Price
Bud	2.50
Miller	2.75
Miller	3.00

Selekcia: σ

Kalkul:

- Nech r je typu r(**X**). Potom

$\sigma_{c(\mathbf{X})}(r) = \{\mathbf{X}: r(\mathbf{X}) \wedge c(\mathbf{X})\}$

Procedurálna definícia: $r_2 := \sigma_c(r_1)$

- r_2 vzniká kopírovaním riadkov r_1, pričom do r_2 sa skopírujú len tie riadky, pre ktoré platí podmienka c

Selekcia: σ

Príklad (Ullman)

Relation sells

Bar	Beer	Price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Miller	3.00

joe_menu := $\sigma_{Bar=\text{“Joe's”}}$(sells)

Bar	Beer	Price
Joe's	Bud	2.50
Joe's	Miller	2.75

Kartézsky súčin: $\times$

Kalkul:

- Nech r_1 je typu $r_1(\mathbf{X})$ a r_2 je typu $r_2(\mathbf{Y})$, pričom $\mathbf{X} \cap \mathbf{Y} = \emptyset$. Potom $r_1 \times r_2 = \{[\mathbf{X}, \mathbf{Y}]: r_1(\mathbf{X}) \wedge r_2(\mathbf{Y})\}$

Procedurálna definícia: $r_3 := r_1 \times r_2$

- r_3 vzniká kopírovaním všetkých dvojíc riadkov r_1 a r_2
- Spoločné atribúty v r_1 a r_2 jednoducho nedovolíme, aby sme sa vyhli problémom v pomenovaní atribútov r_3. (Vlastne môžeme dovoliť aj spoločné atribúty, ale vo výsledku potom musíme dôsledne používať prefixy atribútov—aby bolo jasné, z ktorej relácie pochádzajú.)

Kartézsky súčin: $\times$

Príklad (Ullman)

$$r3 := r1 \times r2$$

r1(

A,	B
1	2
3	4

)

r2(

B,	C
5	6
7	8
9	10

)

r3(

A,	r1.B,	r2.B,	C
1	2	5	6
1	2	7	8
1	2	9	10
3	4	5	6
3	4	7	8
3	4	9	10

)

Join (theta-join): $\bowtie_c$

Kalkul:

- Nech r_1 je typu $r_1(\mathbf{X})$ a r_2 je typu $r_2(\mathbf{Y})$, pričom $\mathbf{X} \cap \mathbf{Y} = \emptyset$. Potom $r_1 \bowtie_{c(\mathbf{X},\mathbf{Y})} r_2 = \{[\mathbf{X}, \mathbf{Y}]: r_1(\mathbf{X}) \wedge r_2(\mathbf{Y}) \wedge c(\mathbf{X}, \mathbf{Y})\}$

Procedurálna definícia: $r_3 := r_1 \bowtie_{c(\mathbf{X},\mathbf{Y})} r_2$

- r_3 vzniká kopírovaním všetkých dvojíc riadkov r_1 a r_2, pričom do r_3 sa skopírujú len tie dvojice, pre ktoré platí podmienka c
- Spoločné atribúty v r_1 a r_2 jednoducho nedovolíme, aby sme sa vyhli problémom v pomenovaní atribútov r_3. (Vlastne môžeme dovoliť aj spoločné atribúty, ale vo výsledku potom musíme dôsledne používať mená relácií ako prefixy atribútov.)
- Theta-join je selekcia aplikovaná na kartézsky súčin

Join (theta-join): $\bowtie_c$

Príklad (Ullman)

sells(

Bar,	Beer,	Price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Coors	3.00

)

bars(

Name,	Addr
Joe's	Maple St.
Sue's	River Rd.

)

bar_info := sells $\bowtie_{sells.bar\ =\ bars.name}$ bars

bar_info(

Bar,	Beer,	Price,	Name,	Addr
Joe's	Bud	2.50	Joe's	Maple St.
Joe's	Miller	2.75	Joe's	Maple St.
Sue's	Bud	2.50	Sue's	River Rd.
Sue's	Coors	3.00	Sue's	River Rd.

)

Premenovanie: ρ

Kalkul:

• Bežná substitúcia používaná matematikmi. Substituovať možno nielen mená atribútov, ale aj meno relácie

Procedurálna definícia: $r_2 := \rho_{r_2(\mathbf{Y})} (r_1)$

• r_2 je kópiou r_1, len sa (možno) volá inak a jej (niektoré) atribúty sa volajú inak

Konkrétna syntax operátora P nie je príliš dôležitá. Avšak musí z nej byť jasné, ktoré meno je pôvodné a ktoré meno je nové. Napríklad, pre reláciu lubi(Pijan, Alkohol) sa $\rho_{\text{Ochmelka := Pijan}}$ (lubi) chápe rovnako ako $\rho_{\text{Ochmelka} \leftarrow \text{Pijan}}$ (lubi) alebo $\rho_{\text{Ochmelka, Alkohol}}$ (lubi)

Premenovanie: ρ

Príklad (Ullman)

bars(

Name,	Addr
Joe’s	Maple St.
Sue’s	River Rd.

)

$\rho_{r(Bar,\ Addr)}$ bars

r(

Bar,	Addr
Joe’s	Maple St.
Sue’s	River Rd.

)

Natural join: $\bowtie$

Kalkul:

• Nech r_1 je typu $r_1(\mathbf{X}, \mathbf{Z})$ a r_2 je typu $r_2(\mathbf{Y}, \mathbf{Z})$, pričom **Z** sú spoločné atribúty r_1 a r_2. Potom

$r_1 \bowtie r_2 = \{[\mathbf{X, Y, Z}]: r_1(\mathbf{X}, \mathbf{Z}) \wedge r_2(\mathbf{Y, Z})\}$

Procedurálna definícia: $r_3 := r_1 \bowtie r_2$

• r_3 vzniká kopírovaním všetkých dvojíc riadkov r_1 a r_2, pričom spoločné atribúty (atribúty s rovnakým menom) sú testované na rovnosť a sú kopírované iba raz

• Natural join sa dá vyjadriť pomocou premenovania, theta-joinu a projekcie. Často je však prirodzené zlúčiť dve relácie do jednej na základe **rovnosti spoločných atribútov**

Natural join: ⋈

Príklad (Ullman)

sells(

Bar,	Beer,	Price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Coors	3.00

)

bars(

Name,	Addr
Joe's	Maple St.
Sue's	River Rd.

)

$$\text{bar\_info} := \text{sells} \bowtie \rho_{\text{bars(Bar, Addr)}}(\text{bars})$$

bars.Name bolo treba premenovať na **bars.Bar**

bar_info(

Bar,	Beer,	Price,	Addr
Joe's	Bud	2.50	Maple St.
Joe's	Milller	2.75	Maple St.
Sue's	Bud	2.50	River Rd.
Sue's	Coors	3.00	River Rd.

)

Tri formy notácie výrazu: bary, ktoré buď sídlia na Maple St., alebo predávajú Bud za menej ako 3$ (Ullman)

Výraz s operátormi

$\pi_{Name}(\sigma_{Addr\,=\,\text{“Maple St.”}}(bars)) \cup$
$\rho_{Name:=Bar}(\pi_{Bar}(\sigma_{Price<3\ AND\ Beer=\text{“Bud”}}(sells)))$

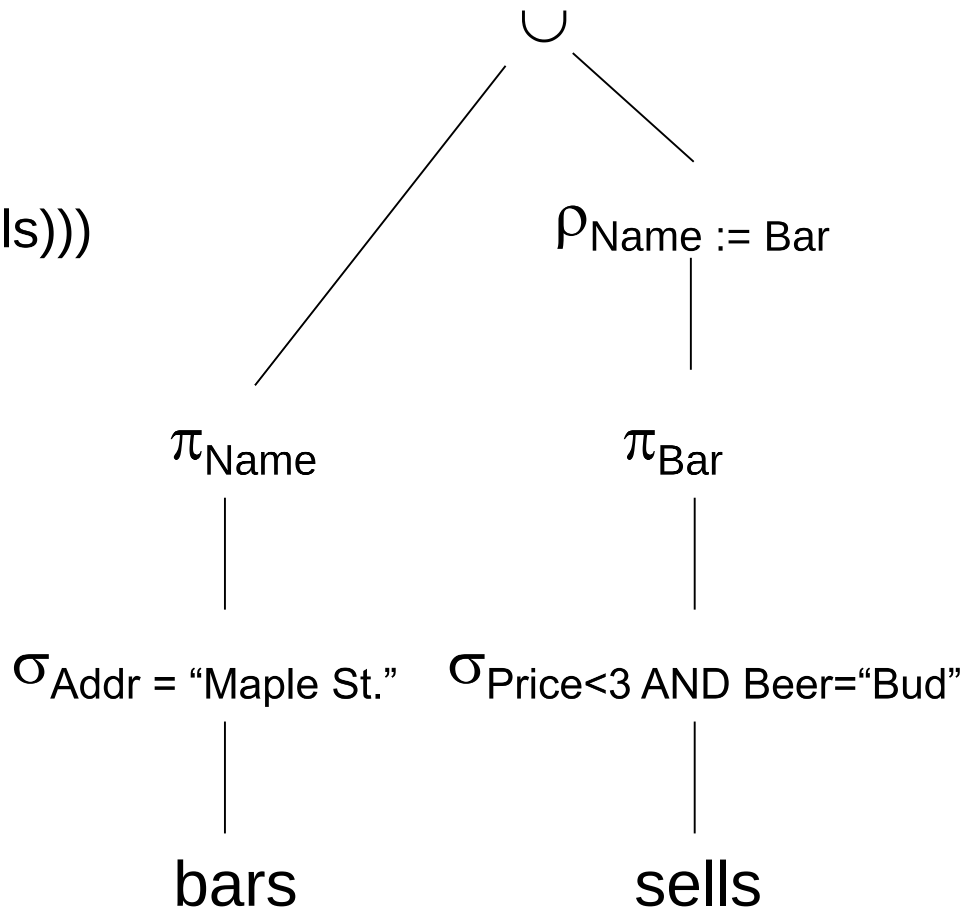
Postupnosť priradení

$r := \pi_{Name}(\sigma_{Addr\,=\,\text{“Maple St.”}}(bars);$

$s := \rho_{Name:=Bar}($
$\pi_{bar}(\sigma_{Price<3\ AND\ Beer=\text{“Bud”}}(sells)));$

$t := r \cup s$

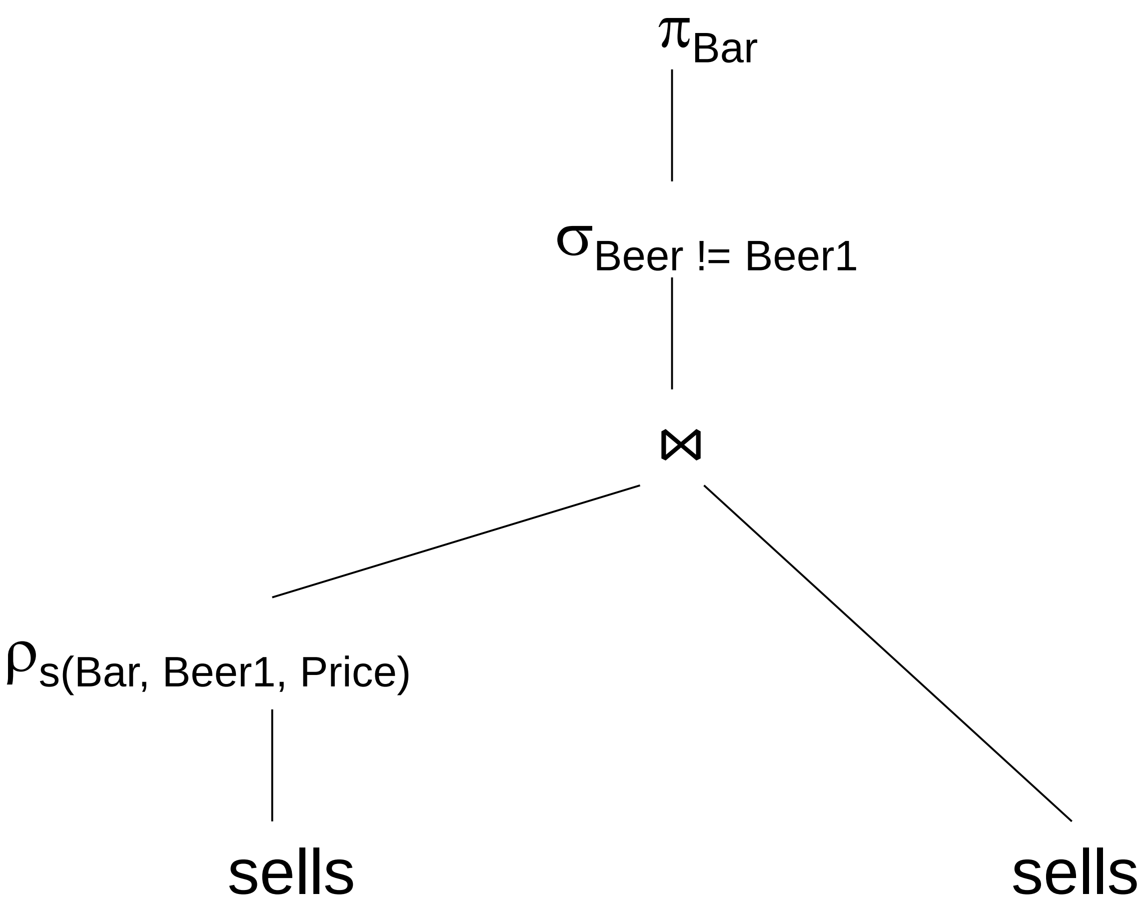
Strom výrazu



Ekvivalentné zápisy výrazov relačnej algebry

Ešte príklad (Ullman): bary, ktoré predávajú (aspoň) dva rôzne druhy piva za rovnakú cenu

$\pi_{Bar}(\sigma_{Beer\ !=\ Beer1}(\rho_{s(Bar,\ Beer1,\ Price)}(sells) \bowtie sells)))$



Niektoré zákony (množinovej) relačnej algebry

- Prirodzené spojenie (natural join) a zjednotenie sú komutatívne, asociatívne a idempotentné
- Platia distributívne zákony
  - $r \bowtie (s \cup t) = (r \bowtie s) \cup (r \bowtie t)$
  - $r \bowtie (s - t) = (r \bowtie s) - (r \bowtie t)$
- Ak $\mathbf{Y} \subseteq \mathbf{X}$, tak potom $\pi_{\mathbf{Y}}\, \pi_{\mathbf{X}}(r) = \pi_{\mathbf{Y}}(r)$
- Ak podmienka c neobsahuje atribúty s, tak potom $\sigma_c(r \times s) = \sigma_c(r) \times s$
- Ak podmienka c_{rs} obsahuje atribúty r aj s, podmienka c_r obsahuje len atribúty r, a podmienka c_s obsahuje len atribúty s, tak potom $\sigma_{c_{rs} \wedge c_r \wedge c_s}(r \times s) = \sigma_{c_r}(r) \bowtie_{c_{rs}} \sigma_{c_s}(s)$

Optimalizácia na úrovni relačnej algebry: príklad

dodava			
Firma	**Vyrobok**	**Cena**	**Lehota**

firmy	
Firma	**Mesto**

objednavky	
Klient	**Vyrobok**

Ktorý klient objednal výrobok, ktorý vie dodať niektorá firma z Nuernberg?

SELECT o.Klient

FROM objednavky o, dodava d, firmy f

WHERE f.Mesto = ‘nuernberg’ **and** f.Firma = d.Firma

and d.Vyrobok = o.Vyrobok

Optimalizácia na úrovni relačnej algebry: príklad

Ktorý klient objednal výrobok, ktorý vie dodať niektorá firma z Nuernberg?

dodava

Firma	**Vyrobok**	**Cena**	**Lehota**
vobis	pc386	2000	4
quelle	pc386	1900	9
vobis	pc486	2900	4
escom	pc486	3000	5
vobis	pc586	5000	7
escom	pc586	5900	9
vobis	hp4l	1400	6
vobis	hddisk	13	0
escom	hddisk	12	0
quelle	cdrom	400	4

dodavatelia

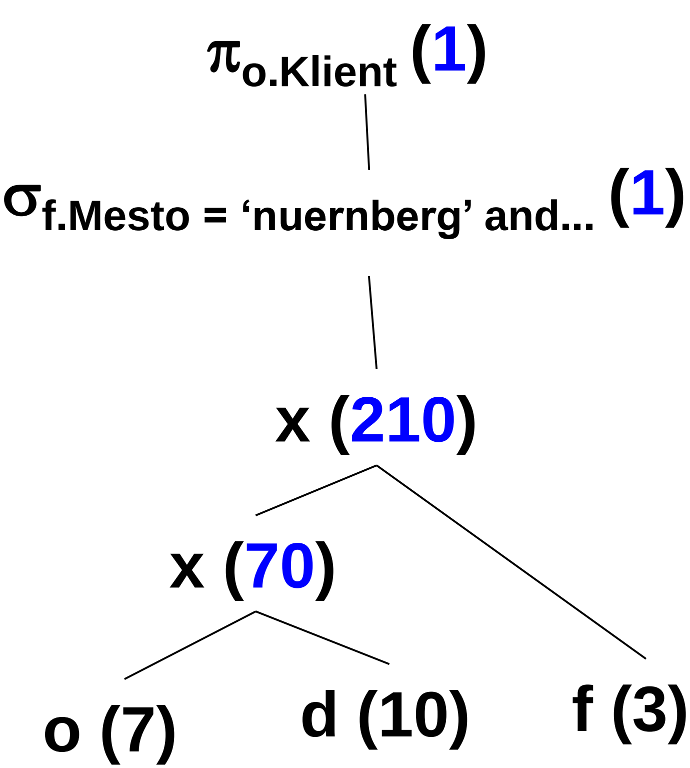
Firma	**Mesto**
vobis	ulm
escom	ulm
quelle	nuernberg

objednavky

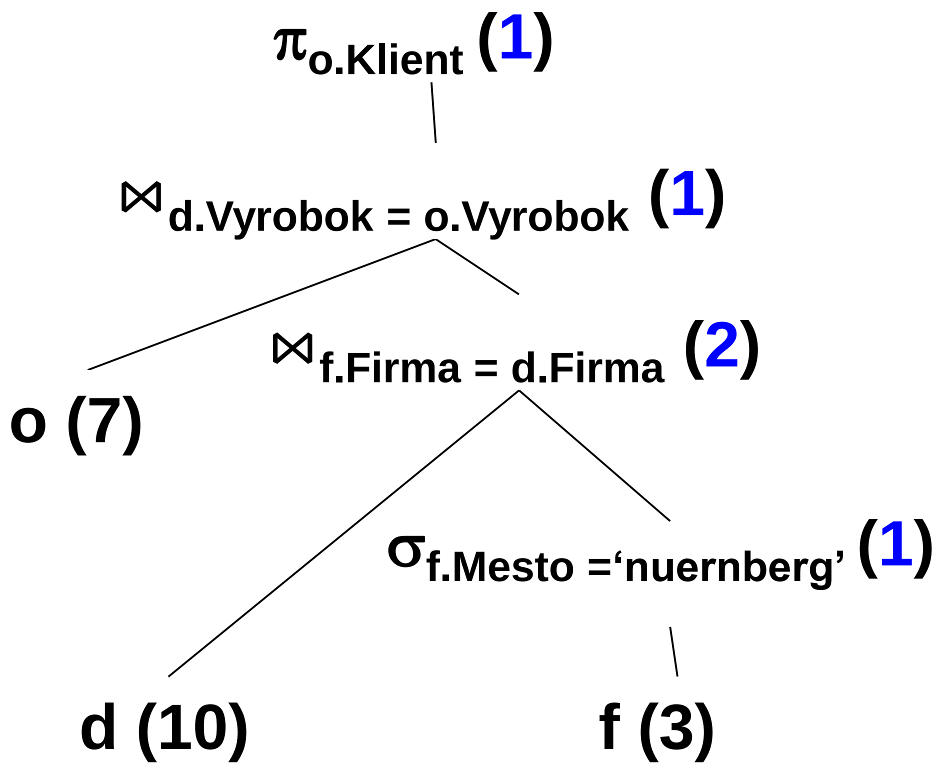
Klient	**Vyrobok**
meier	pc486
meier	hddisk
reich	pc586
reich	hp4l
reich	hddisk
arm	pc386
arm	hddisk

Optimalizácia na úrovni relačnej algebry: príklad

SELECT o.Klient
FROM objednavky o, dodava d, firmy f
WHERE f.Mesto = ‘nuernberg’ **and** f.Firma = d.Firma
and d.Vyrobok = o.Vyrobok



Medzivýsledok: **282 riadkov**



Medzivýsledok: **5 riadkov**

Výpočet jednoduchého SELECT

SELECT $A_1, ..., A_n$
FROM $r_1, ..., r_m$
WHERE c

Projekcia selekcie kartézskeho súčinu

$\pi_{A_1, ..., A_n}\sigma_c(r_1 \times ... \times r_m)$

Projekcia joinu

$\pi_{A_1, ..., A_n}(r_1 \bowtie_c ... \bowtie_c r_m)$

Toto zhruba vyjadruje „kanonický" výpočet výsledku príkazu SELECT (ktorý sa môže ďalej optimalizovať). Zatiaľ sme neriešili napr. spracovanie **duplikátov**

Multimnožiny (bags)

- **Multimnožina (bag)** je množina s duplikátmi. Napríklad {1, 2, 3, 1, 2} je multimnožina. Aj {1, 2, 3} je multimnožina. Každá množina je multimnožinou, ale nie nutne naopak
- SQL počíta nad multimnožinami. Dôvodom je snaha ušetriť, eliminácia duplikátov je rovnako zložitá ako triedenie. (Na druhej strane, skutočne sa šetrí, keď je duplikátov veľa?)
- Relačná algebra počíta nad multimnožinami
- Operátory relačnej algebry sa dajú definovať aj pre multimnožiny
  - *Zjednotenie, prienik a rozdiel* treba popraviť (pre UNION **ALL**, ...)
  - *Projekcia* pre multimnožiny neeliminuje duplikáty
  - Selekcia, kartézsky súčin a joiny sú definované ako predtým (podľa tabuľkovej sémantiky)

Multimnožiny (bags)

Príklad (Ullman)

r(

A,	B
1	2
5	6
1	2

)

$\sigma_{A<3 \wedge B<4}$ (r) =

A	B
1	2
1	2

Multimnožiny (bags)

Príklad (Ullman)

r(

A,	B
1	2
5	6
1	3

)

$\pi_A (r) =$

A
1
5
1

Multimnožiny (bags)

Príklad (Ullman)

r(

A,	B
1	2
5	6
1	2

)

s(

B,	C
3	4
7	8

)

$r \times s =$

A	r.B	s.B	C
1	2	3	4
1	2	7	8
5	6	3	4
5	6	7	8
1	2	3	4
1	2	7	8

Multimnožiny (bags)

Príklad (Ullman)

r(

A,	B
1	2
5	6
1	2

)

s(

B,	C
3	4
7	8

)

$r \bowtie_{r.B<s.B} s =$

A	r.B	s.B	C
1	2	3	4
1	2	7	8
5	6	7	8
1	2	3	4
1	2	7	8

Multimnožiny (bags)

Zjednotenie: multimnožiny sa „zreťazia“.

Príklad: $\{1, 2, 1\} \cup \{1, 1, 2, 3, 1\} = \{1, 1, 1, 1, 1, 2, 2, 3\}$

Prienik: vo výsledku sa prvok objaví toľkokrát, koľkokrát je minimum jeho výskytu v operandoch

Príklad: $\{1, 2, 1, 1\} \cap \{1, 2, 1, 3\} = \{1, 1, 2\}$

Rozdiel: vo výsledku sa prvok objaví toľkokrát, koľkokrát sa vyskytuje v prvom operande mínus koľkokrát sa vyskytuje v druhom operande (samozrejme, aspoň nulakrát)

Príklad: $\{1, 2, 1, 1\} – \{1, 2, 3\} = \{1, 1\}$

Multimnožiny (bags)

Pozor, **nie všetky vlastnosti operácií s množinami sú zachované pre multimnožiny**!

Príklad (Ullman): zjednotenie množín je idempotentné (t.j. $s \cup s = s$), ale zjednotenie multimnožín nie je

Ďalšie operátory relačnej algebry

- Eliminácia duplikátov: δ
- (Triedenie: T)
- OUTERJOIN
- Grupovanie a agregácia: Γ

Eliminácia duplikátov: δ

- $r_2 := \delta(r_1)$

r_2 je kópiou r_1, ale bez duplikovaných riadkov

Príklad (Ullman)

r = (

A	B
1	2
3	4
1	2

)

$\delta(r)$ =

A	B
1	2
3	4

Triedenie: T

- $\mathrm{T}_{X_1,\ldots,X_n}(r)$

je **zoznam**, ktorý vznikol utriedením r najprv podľa X_1, potom podľa X_2, ..., nakoniec podľa X_N (rovnosti sa riešia náhodne). Ak je pred niektorým atribútom šípka nadol, tak sa triedi zostupne, inak sa triedi vzostupne

Príklad (Ullman)

r = (

A	B
1	2
3	4
5	2

)

$\mathrm{T}_B (r)$ = [(5, 2), (1, 2), (3, 4)]

Full join: OUTERJOIN

- $r_3 := r_1$ OUTERJOIN r_2

Full join je podobný theta-joinu, ale do výsledku navyše pribudnú riadky z r_1 a r_2, ktoré sa s ničím nespájajú. Chýbajúce hodnoty v týchto riadkoch sa doplnia špeciálnymi hodnotami **null**

Príklad (Ullman)

r_1 = (

A	B
1	2
4	5

)

r_2 = (

B	C
2	3
6	7

)

[1, 2] sa v $r_1 \bowtie r_2$ spája s [2, 3], ale [4, 5] a [6, 7] sa nespájajú s ničím

r_1 OUTERJOIN r_2 =

A	B	C
1	2	3
4	5	null
null	6	7

Grupovanie a agregácia

$r_2 := \Gamma_X(r_1)$

Vo vektore **X** môžu byť použité

- **grupovacie atribúty** (atribúty z r_1)
- **agregácie** tvaru AGG(Y), kde Y je atribútom r_1 (tzv. agregovaný atribút) a AGG je niektorá z agregačných funkcií SUM, COUNT, AVG, STDEV, MAX, MIN

Operátor $\Gamma_X(r_1)$

1. vyrobí z relácie r_1 skupiny, pričom riadky v každej zo skupín sa zhodujú vo všetkých hodnotách grupovacích atribútov
2. vypočíta všetky agregácie, pre každú skupinu zvlášť

Výsledkom je tabuľka, v ktorej každej skupine prináleží jeden riadok

Grupovanie a agregácia

Príklad (Ullman) r = (

A	B	C	D
1	2	3	1
4	5	6	2
1	2	5	3

)

$\Gamma_{A,\ B,\ AVG(C)}$ (r) = ??

1.krok: zgrupuj r podľa A a B :

A	B	C	D
1	**2**	3	1
1	**2**	5	2
4	**5**	6	3

2.krok: vypočítaj AVG(C) pre každú grupu:

A	B	AVG(C)
1	2	4
4	5	6

Grupovanie a agregácia presnejšie

$\Gamma_{A_1, A_2, \ldots, A_n}$ (r) je agregačný operátor aplikovaný na r:

• A_i je buď **grupovací atribút** alebo **agregácia** [SUM(A_i) | COUNT(A_i) | AVG(A_i) | STDEV(A_i) | MIN(A_i), MAX(A_i)] (kde A_i je **agregovaný atribút**). Nech **G** je množina grupovacích atribútov a **A** je množina agregovaných atribútov. (r môže obsahovať aj iné atribúty, t.j. také, ktoré nepatria do **G** ani do **A**.)

• Medzivýsledkom je relácia r´, ktorá vzniká projekciou r na množinu atribútov **G** s následným odstránením duplikátov: r´ := $\delta(\Pi_{\mathbf{G}}(r))$. **Jeden riadok r´ zodpovedá skupine riadkov v r**

• Výsledkom je relácia r´ rozšírená o výsledky agregácií, pričom hodnota agregácie v riadku výsledku sa vypočíta aplikáciou agregačnej funkcie na hodnoty agregovaného atribútu v zodpovedajúcej skupine riadkov v r

Rozšírenie jazykov o agregáciu: SQL

Syntax a sémantika SELECT s GROUP BY a HAVING:

SELECT <S_attr> ***5***

FROM $r_1, r_2, \ldots, r_n$ ***1***

WHERE <w_cond> ***2***

GROUP BY <G_attr> ***3***

HAVING <h_cond> ***4***

$$\pi_{S\_attr}(\sigma_{h\_cond}\ (\Gamma_{G\_attr,\ AGG(Attr)}(\sigma_{w\_cond}(r_1 \times r_2 \times \ldots \times r_n))))$$

5 4 3 2 1