

Operasi Relasional Basis Data

D. Sinaga, M.Kom

Aljabar Relasional

- ▶ Adalah sebuah bahasa query prosedural yang terdiri dari sekumpulan operasi dimana masukkannya adalah satu atau dua relasi dan keluarannya adalah sebuah relasi baru sebagai hasil dari operasi tersebut.
- ▶ **Perintah dasar dari Operasi dalam Aljabar Relasional adalah select, project, union, set difference, dan cartesian product.** namun selain dari perintah dasar dari operasi aljabar Relasional ada beberapa tambahan operasi seperti **set intersection, natural join, division dan theta join.**

Fungsi dari Operasi-operasi Dasar Aljabar Relasional sebagai berikut :

Operasi-operasi dasar

1. Select (σ)
2. Projection (π)
3. Union (\cup)
4. Intersection (\cap)
5. Set difference (-)
6. Rename (ρ)

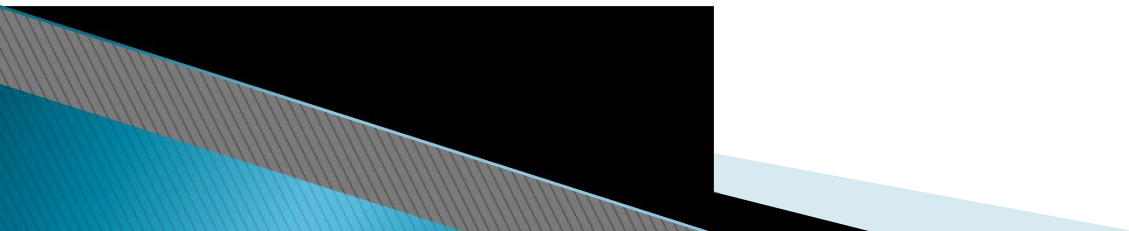
Operasi lainnya

1. Cross / cartesian product (\times)
2. Join (\bowtie)
3. Division (\div)
4. sum, average, min, max

- ▶ **Operasi Select** berfungsi untuk menyeleksi tuple-tuple yang memenuhi predikat yang diberikan dari sebuah tabel relasi. dan simbol yang di gunakan adalah simbol sigma “ σ ” simbol ini digunakan untuk menunjukkan operasi select.
- ▶ Contoh penggunaan operasi select :
- ▶ Misalkan kita mau mencari hasil NILAI berdasar $\sigma_{NPM = 10296832}$ (NILAI) maka hasil select yang keluar

Jenis Operasi

- ▶ Operasi unary terdiri dari selection, projection. Disebut operasi unary karena dapat digunakan hanya pada satu tabel.
- ▶ Operasi binary terdiri dari union, intersection, set difference, cartesian product, join dan division. Disebut operasi binary karena memerlukan sepasang tabel.



Operasi Selection (σ)

- ▶ Operasi yang digunakan untuk memilih tupel–tupel yang memenuhi suatu predikat dapat menggunakan operator perbandingan ($=, \neq, >, \geq, <, \leq$) pada predikat.
- ▶ Beberapa predikat dapat dikombinasikan menjadi predikat majemuk menggunakan penghubung AND (\wedge) dan OR (\vee). Contoh :
- ▶ Query : tampilkan jenis film dari tabel film yang jenisnya adalah “ action “
- ▶ aljabar relasional : $\sigma_{\text{jenis}=\text{“action”}}(\text{film})$

► Tabel : film

kode_film	jenis	judul	jml_keping	jml_film
A01	action	Spiderman	2	3
A02	action	Spiderman 2	2	5
D01	drama	Love Story	1	3
H01	horor	Evil Death	3	2

kode_film	jenis	judul	jml_keping	jml_film
A01	action	Spiderman	2	3
A02	action	Spiderman 2	2	5

Projection (π)

- ▶ Operasi yang digunakan untuk memilih kolom dalam satu tabel:
- ▶ Notasi : $\pi_{A1,A2, \dots, An}(t)$ dimana $A1, A2, \dots, An$ adalah nama atribut dan t adalah nama tabel.
- ▶ Hasilnya : suatu tabel dengan atribut yang tercantum pada daftar nama atribut pada operasi. Contoh :
- ▶ Query : tampilkan kode_film, jenis dan judul dari tabel film
- ▶ Aljabar relasionalnya : $\pi_{kode_film, jenis, judul}(film)$

Hasilnya :

kode_film	jenis	judul	jml_keping	jml_film
A01	action	Spiderman	2	3
A02	action	Spiderman 2	2	5
D01	drama	Love Story	2	3
H01	horor	Evil Death	2	2

- Hasilnya :

kode_film	jenis	judul
A01	action	Spiderman
A02	action	Spiderman 2
D01	drama	Love Story
H01	horor	Evil Death

Union (\cup)

- Notasi : $r \cup s$, menghasilkan suatu tabel baru yg elemennya barisnya merupakan elemen dari r dan s , tidak ada duplikasi data
- Notasi : $r \cup s = \{ x \mid x \in r, x \in s \}$

• tabel r

A	B	C
aa	1	7
bb	5	7
bb	12	7
dd	23	10



• tabel s

A	B	C
aa	5	7
cc	4	4
dd	10	7
aa	6	10

• $\pi_A(r) \cup \pi_A(s)$

A
aa
bb
dd
cc

Cartesian (X)

- Proses yang menghasilkan tabel hasil perkalian dua tabel.
- Notasi : $r \times s = \{ (x,y) \mid x \in r \text{ dan } y \in s \}$
- Query : tampilkan a,b dari (r) dan c,d dari (s)
- Aljabar relationalnya : $\pi_{a, b, c, d} (r \times s)$

• tabel r :			maka r x s :				
	A	B		A	B	C	D
	a	1		a	1	a	6
	b	3		a	1	b	5
• tabel s :				a	1	c	12
	C	D		b	3	a	6
	a	6		b	3	b	5
	b	5		b	3	c	12
	c	12					

Menggabungkan dengan ekspresi aljabar lainnya

- Tabel film :

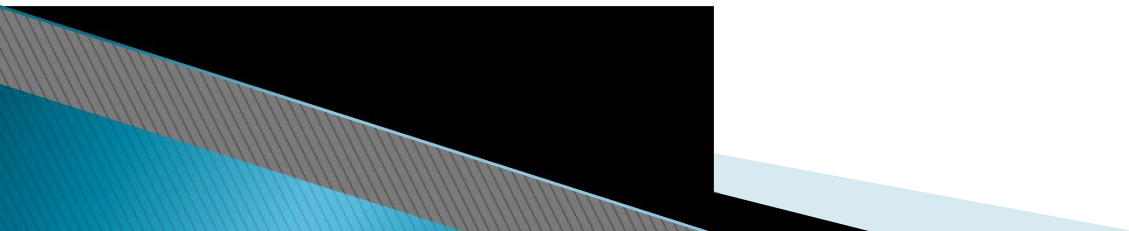
kode_film	jenis	judul	jml_keping	jml_film
A01	action	Spiderman	2	3
A02	action	Spiderman 2	2	5
D01	drama	Kabayan	2	3
H01	horor	Scream	2	2

- 1. Mengekstraksi film yang jumlah filmnya lebih besar dari 3
- Aljabar relasionalnya : $\pi_{\text{kode_film, judul, jml_film}}(\sigma_{\text{jml_film} > 3}(\text{film}))$

kode_film	judul	jml_film
A02	Spiderman 2	5

Latihan :

- ▶ Tampilkan dengan query dan aljabar relational kolom : a,b,c,d dari tabel s dan r yang $b=1$.



Difference (--)

- Operasi untuk mendapatkan tabel baru dari sebuah relasi dimana elemen barisnya terdapat di r tetapi tidak ada di s. r dan s harus memiliki jumlah atribut yang sama.
- Notasi : $r-s = \{x \mid x \in r \text{ dan } x \notin s\}$
- Query : tampilkan a,b,c dari tabel r dan s
- Aljabar relasionalnya : $\pi a(r) - \pi a(s)$

● tabel r

A	B	C
aa	1	7
bb	5	7
bb	12	7
dd	23	10

● tabel s

A	B	C
aa	5	7
cc	4	4
dd	10	7
aa	6	10

● $\pi_A(r) - \pi_A(s)$

A
bb

Rename (ρ)

- Operasi untuk menyalin tabel lama ke dalam tabel yang baru.
- Query : buatlah tabel baru dengan nama tb dari tabel r dimana $b=5$
- Aljabar relasionalnya : $\rho_{tb}(\sigma_{b=5}(r))$
- Akan terbentuk sebuah tabel baru bernama tb dengan isi $b=5$;

tabel r

A	B	C
aa	1	7
bb	5	7
bb	12	7
dd	23	10

Set intersection (\cap)

- Operasi binary yang digunakan untuk membentuk sebuah relasi baru dengan baris(tupel) yang berasal dari kedua tabel yg dihubungkan.
- Notasi : $r \cap s = r - (r - s)$ atau $r \cap s = s - (s - r)$
- Aljabar relasionalnya : $\pi_a(r) \cap \pi_a(s)$

tabel r

A	B	C
aa	1	7
bb	5	7
bb	12	7
dd	23	10

• tabel s

A	B	C
aa	5	7
cc	4	4
dd	10	7
aa	6	10

\cap

Division (\div)

- ▶ Operasi division berfungsi untuk query yang memasukkan frase “untuk semua/seluruh”. Simbol “ \div ” digunakan untuk menunjukkan operasi division.
- ▶ Misalkan terdapat 2 tabel relasi bernama Mengajar dan Dosen
- ▶ dibawah :

Query : Tampilkan nid, hari, waktu (dari relasi Mengajar) dan nid (dari relasi Dosen) dimana dosen yang jenis kelaminnya 'Pria' dan lakukan devision pada kedua relasi tersebut.

Aljabar relasional: $\pi_{nid, hari, waktu} (Mengajar) \div (\pi_{nid} (\sigma_{jkelamin='Pria'} (Dosen)))$

$\pi_{nid, hari, waktu} (Mengajar)$

nid	hari	waktu
00001	Rabu	8:00
00001	Senin	8:00
00002	Jumat	8:00
00002	Jumat	10:00
95001	Kamis	8:00
95001	Senin	8:00
98002	Rabu	8:00
98002	Selasa	10:00
99001	Senin	8:00
99001	Selasa	8:00

$\pi_{nid} (\sigma_{gajipokok > 1300000} (Dosen))$

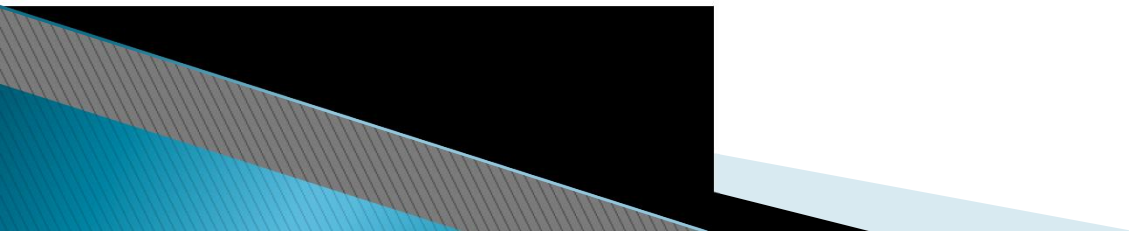
nid
00001
95001

Hasil akhir adalah:

nid
98002
99001
99001

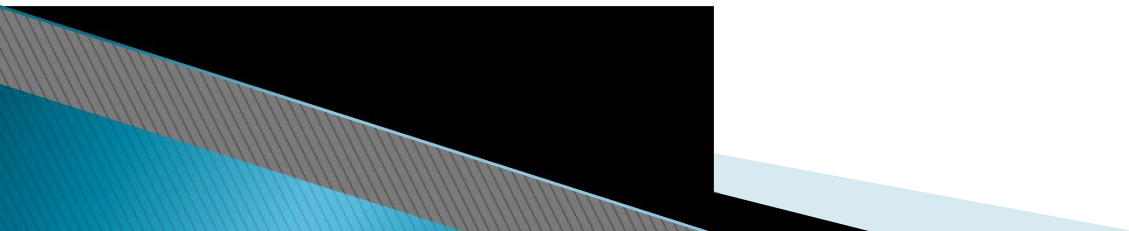
Optimasi Query

- ▶ **Optimasi Query** merupakan proses untuk menganalisa query dan menentukan sumber-sumber apa saja yang digunakan oleh query tersebut dan apakah pengguna dari sumber tersebut dan apakah pengguna tersebut di kurangi tanpa merubah output.
- ▶ Ada tiga aspek dasar yang mempengaruhi optimasi query yaitu : Search Space, Cost Model dan Search Strategy



Tujuan Optimasi Query

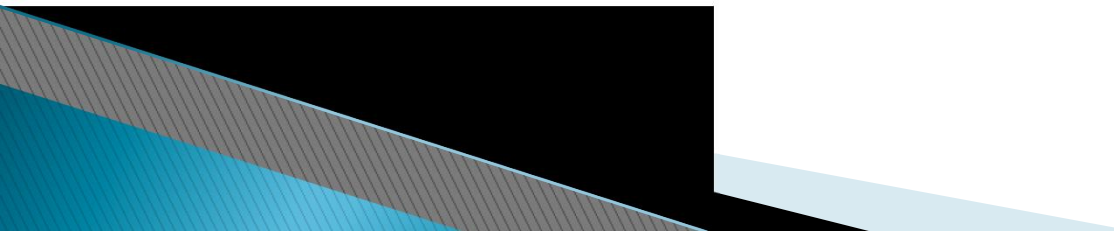
- ▶ Dalam tujuannya optimasi query digunakan untuk meminimumkan waktu proses, I/O, pengguna memory dan meminimumkan total waktu pada proses query.
- ▶ Menurut Chanowich (2001) ada 2 cara pendekatan optimasi yang digunakan untuk saat ini yaitu :
 - Heuristik atau Rule-Based
 - Cost-Based.



1. Heuristik atau Rule-Based

Optimasi untuk jenis ini mentransformasikan query yang akan meningkatkan kinerja eksekusi yaitu :

mereduksi jumlah baris dengan melakukan operasi selection.

- ▶ mereduksi jumlah atribut dengan melakukan operasi projection.
 - ▶ mengkonversi query dengan banyak join menjadi query dengan banyak subquery.
 - ▶ melakukan operasi selection dan join yang paling kecil keluarnya sebelum operasi lain.
- 

2. Cost-Based

- ▶ Optimasi untuk jenis ini dipergunakan dari beberapa alternatif untuk dipilih menjadi salah satu cost yang terendah. selain itu teknik ini juga mengoptimalkan urutan join terbalik pada relasi-relasi R_1 sampai R_n . Pada teknik ini juga dipergunakan untuk mendapatkan pohon left-deep join agar menghasilkan relasi pada node sebelah kanan.
- ▶ Selain Teknik tersebut juga masih ada teknik lainnya yaitu :
 - Join Ordering merupakan suatu aspek penting dalam optimasi query
 - Algoritma Sistem R merupakan optimasi query statis berdasarkan exhaustive search
 - Algoritma Ingres merupakan algoritma optimasi dinamis yang memecah query kalkulus kebagian yang lebih kecil

- ▶ The sql query becomes faster if you use the actual columns names in SELECT statement instead of than '*'.
- ▶ **For Example:** Write the query as :
SELECT id, first_name, last_name, age, subject FROM student_details;
Instead of:
SELECT * FROM student_details;

- ▶ **HAVING** clause is used to filter the rows after all the rows are selected. It is just like a filter. Do not use **HAVING** clause for any other purposes.

For Example: Write the query as

```
SELECT subject, count(subject)
FROM student_details
WHERE subject != 'Science'
AND subject != 'Maths'
GROUP BY subject;
```

Instead of:

```
SELECT subject, count(subject)
FROM student_details
GROUP BY subject
HAVING subject!= 'Vancouver' AND subject!= 'Toronto';
```

Sometimes you may have more than one subqueries in your main query. Try to minimize the number of subquery block in your query.

For Example: Write the query as

```
SELECT name  
FROM employee  
WHERE (salary, age ) = (SELECT MAX (salary), MAX (age)  
FROM employee_details)  
AND dept = 'Electronics';
```

Instead of:

```
SELECT name  
FROM employee  
WHERE salary = (SELECT MAX(salary) FROM  
employee_details)
```

