Institut Informatika & Bisnis DARMAJAYA
Yayasan Alfian Husin

SAINS DATA
Institut Informatika & Bisnis Darmajaya

Kampus Merdeka INDONESIA JAYA

MERDEKA BELAJAR

DATA SCIENCE DARMAJAYA
"YOUR BEST FUTURE IN DATA"

MEETING: [7 & 8]

THE RELATIONAL LANGUAGE (Cont'd)

BY: HENDRA KURNIAWAN

[12-09-2023] | [KODE MK: SD23404 | MATAKULIAH: PENGANTAR BASIS DATA | SKS: 2/2 | VERSI: 01

The Relational Language (Cont'd)

1. Relational Algebra.
2. Relational Calculus.

Learning Objectives

1. The meaning of the term “relational completeness”.
2. How to form queries in the relational algebra.
3. How to form queries in the tuple relational calculus.
4. How to form queries in the domain relational calculus.

Relational Algebra (Additional Operator)

- Additional Operations
  - Set intersection
  - Natural join
  - Aggregation
  - Outer Join
  - Division
- All above, other than aggregation, can be expressed using basic operations we have seen earlier.

Set-Intersection Operation – Example

- Relation *r, s*:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

- $r \cap s$

A	B
α	2

Natural Join Operation – Example

- Relations r, s:

A	B	C	D
α	1	α	a
β	2	γ	a
γ	4	β	b
α	1	γ	a
δ	2	β	b

r

B	D	E
1	a	α
3	a	β
1	a	γ
2	b	δ
3	b	$\in$

s

- r ⋈ s

A	B	C	D	E
α	1	α	a	α
α	1	α	a	γ
α	1	γ	a	α
α	1	γ	a	γ
δ	2	β	b	δ

Natural Join Operation – Example

- Let *r* and *s* be relations on schemas *R* and *S* respectively.
  Then, r s is a relation on schema $R \cup S$ obtained as follows:
  - Consider each pair of tuples t_r from *r* and t_s from *s*.
  - If t_r and t_s have the same value on each of the attributes in $R \cap S$, add a tuple *t* to the result, where
    - *t* has the same value as t_r on *r*
    - *t* has the same value as t_s on *s*

Natural Join Operation – Example

- Example:

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

  $S = (E, B, D)$

  - Result schema = (A, B, C, D, E)
  - $r \bowtie s$ is defined as:

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

Aggregate Functions and Operations

- **Aggregation function** takes a collection of values and returns a single value as a result.

  avg: average value
  min: minimum value
  max: maximum value
  sum: sum of values
  count: number of values

- **Aggregate operation** in relational algebra

$$_{G_1,G_2,\ldots,G_n}\mathcal{G}_{F_1(A_1),F_2(A_2,\ldots,F_n(A_n)}(E)$$

  E is any relational-algebra expression

  - G_1, G_2 …, G_n is a list of attributes on which to group (can be empty)
  - Each F_i is an aggregate function
  - Each A_i is an attribute name

Aggregate Operations - Example

- Relation *r*:

A	*B*	*C*
α	α	7
α	β	7
β	β	3
β	β	10

- $g_{\text{sum(c)}}(\text{r})$

sum(*c*)
27

- Question: Which aggregate operations cannot be expressed using basic relational operations?

Aggregate Operations - Example

- Relation *account* grouped by *branch-name*:

branch_name	*account_number*	*balance*
Perryridge	A-102	400
Perryridge	A-201	900
Brighton	A-217	750
Brighton	A-215	750
Redwood	A-222	700

$$branch\_name\ \mathcal{G}_{\mathbf{sum}(balance)}\ (account)$$

branch_name	**sum**(*balance*)
Perryridge	1300
Brighton	1500
Redwood	700

Aggregate Function

- Result of aggregation does not have a name
  - Can use rename operation to give it a name
  - For convenience, we permit renaming as part of aggregate operation

$$branch\_name\ \mathcal{g}\ \textbf{sum}(balance)\ \textbf{as}\ sum\_balance\ (account)$$

Outer Join

- An extension of the join operation that avoids loss of information.
- Computes the join and then adds tuples form one relation that does not match tuples in the other relation to the result of the join.
- Uses *null* values:
  - *null* signifies that the value is unknown or does not exist
  - All comparisons involving *null* are (roughly speaking) **false** by definition.
    - We shall study precise meaning of comparisons with nulls later

Outer Join - Example

- Relation *loan*

loan_number	*branch_name*	*amount*
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

- Relation *borrower*

customer_name	*loan_number*
Jones	L-170
Smith	L-230
Hayes	L-155

Outer Join - Example

- Join

  loan ⋈ *borrower*

loan_number	*branch_name*	*amount*	*customer_name*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith

- Left Outer Join

  loan ⟕ *borrower*

loan_number	*branch_name*	*amount*	*customer_name*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-260	Perryridge	1700	*null*

Outer Join - Example

- Right Outer Join

  loan ⟖ *borrower*

loan_number	*branch_name*	*amount*	*customer_name*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-155	*null*	*null*	Hayes

- Full Outer Join

  loan ⟗ *borrower*

loan_number	*branch_name*	*amount*	*customer_name*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-260	Perryridge	1700	*null*
L-155	*null*	*null*	Hayes

Division Operation

Notation:

Suited to queries that include the phrase “for all”.

Let r and s be relations on schemas R and S respectively where

$R = (A_1, \ldots, A_m, B_1, \ldots, B_n)$

$S = (B_1, \ldots, B_n)$

The result of r ÷ s is a relation on schema

$R - S = (A_1, \ldots, A_m)$

$$r \div s = \{ t \mid t \in \prod_{R-S}(r) \wedge \forall u \in s (tu \in r) \}$$

Where tu means the concatenation of tuples t and u to produce a single tuple

Division Operation - Example

- Relations *r*, *s*:

A	B
α	1
α	2
α	3
β	1
γ	1
δ	1
δ	3
δ	4
ϵ	6
ϵ	1
β	2

r

B
1
2

s

- $r \div s$:

A
α
β

Another Division Operation - Example

- Relations *r*, *s*:

A	*B*	*C*	*D*	*E*
α	a	α	a	*1*
α	a	γ	a	*1*
α	a	γ	b	*1*
β	a	γ	a	*1*
β	a	γ	b	*3*
γ	a	γ	a	*1*
γ	a	γ	b	*1*
γ	a	β	b	*1*

r

D	*E*
a	*1*
b	*1*

s

- *r* ÷ *s*:

A	*B*	*C*
α	a	γ
γ	a	γ

CONCLUSION

In summary, relational algebra is a fundamental concept in the field of database management that provides a set of operations to manipulate and retrieve data from relational databases. These operations include Selection, which allows for the retrieval of specific rows based on certain conditions; Projection, which focuses on selecting specific columns from a table; Union, which combines data from two or more tables while eliminating duplicates; Intersection, which returns only the common rows between two tables; Difference, which returns the rows unique to one table compared to another; and Rename, which allows for the renaming of tables and columns for clarity and consistency. These operations are crucial for querying and manipulating data in relational databases, forming the basis for many database management systems and SQL queries.

REFERENCES

1. Connoly. T., Begg. Carolyn. 2015. Database System: A Pratical Approach to Design, Implementation, and Management. Sixth Edition. Global Edition. Pearson.

Institut Informatika & Bisnis DARMAJAYA
Yayasan Alfian Husin

SAINS DATA
Institut Informatika & Bisnis Darmajaya

Kampus Merdeka INDONESIA JAYA

MERDEKA BELAJAR

THANK YOU!!

DATA SCIENCE DARMAJAYA “YOUR BEST FUTURE IN DATA”