Databases
Relational algebra

Hans Philippi

Example database inspired by imdb.com

August 1, 2024

Why relational algebra?

- fundamental query language
- based on set theory
- *yardstick*: relational completeness
- *compositional*: a query may be composed from subqueries
- *procedural*: ordering of operations
- *concise*: only five basic operators
- *practical use*: query optimization

Relational model: glossary

Movie			
movid	title	year	rating
tt0469494	There Will Be Blood	2007	8,1
tt0086879	Amadeus	1984	8,4
tt0102926	The Silence of the Lambs	1991	8,6
tt0110413	Léon	1994	8,7
tt0078788	Apocalypse Now	1979	8,5

The *schema* of this *relation (table)* is:

Movie(movid, title, year, rating)

There are five *tuples (records, rows)*

There are four *attributes (fields)*: movid, title, year, rating

There are four *columns*, identified by an attribute, each containing five values

The *degree* of the relation Movie is four

Relational model: subtleties

We should regard these two relations to be identical, because relations are sets

Table1	
A	B
1	game
2	set
3	match

Table2	
A	B
3	match
1	game
2	set

Please note that sets do not contain duplicates

Relational model: subtleties

Should we regard these two relations to be identical?

Table1	
A	B
1	game
2	set
3	match

Table2	
B	A
game	1
set	2
match	3

Relational model: subtleties

Should we regard these two relations to be identical?

Table1	
A	B
1	game
2	set
3	match

Table2	
B	A
game	1
set	2
match	3

- From a practical point of view (query optimization), this is a desirable property
- We can define a tuple as a function from the set of attributes to the set of all possible data values, giving the desired property

Selection

Unary operator: **selection** σ_p — p is selection predicate

Actor		
personid	name	birth_year
nm0000204	Natalie Portman	1982
nm0000288	Christian Bale	1974
nm0000358	Daniel Day-Lewis	1957
nm0000201	Michelle Pfeiffer	1958

$\sigma_{birth\_year<1960}$(Actor)		
personid	name	birth_year
nm0000358	Daniel Day-Lewis	1957
nm0000201	Michelle Pfeiffer	1958

Selection

Unary operator: **selection** with complex predicates

$$\sigma_{(birth\_year > 2000) \wedge (gender = 'female')}(\texttt{Actor})$$

$$\sigma_{(country = 'Netherlands') \vee (country = 'Belgium')}(\texttt{Actor})$$

Complex selection predicates are built with $\wedge, \vee, \neg$ and brackets, but without quantors ($\exists$, $\forall$)

Projection

Unary operator: **projection** π_L — L is projection set

Movie			
movid	title	year	rating
tt0049366	Invasion of the Body Snatchers	1956	7,8
tt0086879	Amadeus	1984	8,4
tt0077745	Invasion of the Body Snatchers	1978	7,4
tt0078788	Apocalypse Now	1979	8,5

$\pi_{title,year}$ (Movie)	
title	year
Invasion of the Body Snatchers	1956
Amadeus	1984
Invasion of the Body Snatchers	1978
Apocalypse Now	1979

Projection

Movie			
movid	title	year	rating
tt0049366	Invasion of the Body Snatchers	1956	7,8
tt0086879	Amadeus	1984	8,4
tt0077745	Invasion of the Body Snatchers	1978	7,4
tt0078788	Apocalypse Now	1979	8,5

π_{title}(Movie)
title
Invasion of the Body Snatchers
Amadeus
Apocalypse Now

Composition of operators

Composition of projection and selection

Actor		
personid	name	birth_year
nm0000204	Natalie Portman	1982
nm0000288	Christian Bale	1974
nm0000358	Daniel Day-Lewis	1957
nm0000201	Michelle Pfeiffer	1958

$\pi_{name,birth\_year}(\sigma_{birth\_year<1960}(\texttt{Actor}))$	
name	birth_year
Daniel Day-Lewis	1957
Michelle Pfeiffer	1958

Union

Binary operator: **union** $\cup$ schema compatibility

Actor		
personid	name	birth_year
nm0000531	Frances McDormand	1957
nm0000233	Quentin Tarantino	1963
nm0000358	Daniel Day-Lewis	1957

Director		
personid	name	birth_year
nm0000759	Paul Thomas Anderson	1970
nm0000941	Kathryn Bigelow	1951
nm0000233	Quentin Tarantino	1963

Union

Binary operator: **union** $\cup$ schema compatibility

Actor ∪ Director		
personid	name	birth_year
nm0000941	Kathryn Bigelow	1951
nm0000531	Frances McDormand	1957
nm0000358	Daniel Day-Lewis	1957
nm0000759	Paul Thomas Anderson	1970
nm0000233	Quentin Tarantino	1963

Difference

Binary operator: **difference** − schema compatibility

Actor − Director		
personid	name	birth_year
nm0000531	Frances McDormand	1957
nm0000358	Daniel Day-Lewis	1957

Intersection

Binary operator: **intersection** $\cap$ schema compatibility

Actor ∩ Director		
personid	name	birth_year
nm0000233	Quentin Tarantino	1963

Cartesian product

Binary operator: **Cartesian product** $\times$

R	
A	B
a	11
b	43

S	
C	D
b	25
c	41
b	21

R × S			
A	B	C	D
a	11	b	25
a	11	c	41
a	11	b	21
b	43	b	25
b	43	c	41
b	43	b	21

Theta-join

Binary operator: **theta-join** $\bowtie_\theta$

θ is matching condition

R	
A	B
a	11
b	43
c	37

S	
C	D
b	55
c	31
b	21

R $\bowtie_\theta$ S			
A	B	C	D
b	43	b	21
c	37	c	31

$$\theta : (R.A = S.C) \wedge (R.B > S.D)$$

Natural join

Binary operator: **natural join** $\bowtie$ default matching condition

R	
A	B
a	11
b	43
c	37

S	
A	D
b	55
c	31
b	21
d	17

R $\bowtie$ S		
A	B	D
b	43	55
b	43	21
c	37	31

Examples

Library schema: [1]

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q1: Give the names of the readers who borrowed at least one book of Dickens

[1] For simplicity, we assume that every title has only one copy in our library

Examples

Library schema:

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q1: Give the names of the readers who borrowed at least one book of Dickens

$$\pi_{name}(Reader \bowtie (Loan \bowtie (\sigma_{author=''Dickens''} Book)))$$

Equivalencce of expressions

Library schema:

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q1: Give the names of the readers who borrowed at least one book of Dickens

$$\pi_{name}(Reader \bowtie (Loan \bowtie (\sigma_{author="Dickens"} Book)))$$

... but what about this one?

$$\pi_{name}(\sigma_{author="Dickens"}(Reader \bowtie Loan \bowtie Book))$$

Examples

Library schema:

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q2: Give the names of the readers who never borrowed a book of Dickens

Examples

Library schema:

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q2: Give the names of the readers who never borrowed a book of Dickens

$$\pi_{name}(Reader \bowtie (Loan \bowtie (\sigma_{author \neq \text{"Dickens"}} Book)))$$

Note that this attempt fails. What does this expression mean?

Examples

Library schema:

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q2: Give the names of the readers who never borrowed a book of Dickens

First step: the completely incorrect answer

$$\pi_{rid}(Loan \bowtie (\sigma_{author="Dickens"} Book))$$

Examples

Library schema:

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q2: Give the names of the readers who never borrowed a book of Dickens

Second step: take the complement of the first step

$$\pi_{rid}(Reader) - \pi_{rid}(Loan \bowtie (\sigma_{author="Dickens"} Book))$$

To project on the names, a final join with Reader is required

Examples

Library schema:

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q3: Give the names of the readers who borrowed only Dickens-books

Examples

Library schema:

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q3: Give the names of the readers who borrowed only Dickens-books

The answer only requires a minor modification of Q2

$$\pi_{rid}(Reader) - \pi_{rid}(Loan \bowtie (\sigma_{author \neq \text{"Dickens"}} Book))$$

Examples

Library schema:

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q4: Give the names of the readers who borrowed all Dickens-books

???

Division

Binary operator: **division** $\div$

T	
A	B
1	1
1	3
1	4
2	2
2	4
6	1
6	3
8	1
8	3
8	4
8	7

U
B
1
3
4

$T \div U$
A
1
8

Examples

Library schema:

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q4: Give the names of the readers who borrowed all Dickens-books

Examples

Library schema:

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q4: Give the names of the readers who borrowed all Dickens-books

$$\ldots \div \pi_{bid}(\sigma_{author=''Dickens''} Book)$$

Examples

Library schema:

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q4: Give the names of the readers who borrowed all Dickens-books

$$\pi_{rid,bid}(Loan) \div \pi_{bid}(\sigma_{author="Dickens"} Book)$$

Examples

Library schema:

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q4: Give the names of the readers who borrowed all Dickens-books

$$Loan \div \pi_{bid}(\sigma_{author="Dickens"} Book)$$

Why does this attempt fail?
And what is the meaning of this expression?

Assignment & renaming

Unary operators: **assignment & renaming**

$$T := < alg\_expr >$$

$$T[A_1, ..., A_n] := < alg\_expr >$$

Examples

Unary operators: **renaming on the fly**

$$\rho(T)(< alg\_expr >)$$

$$\rho(T, A_1, ..., A_n)(< alg\_expr >)$$

Examples

assignment & renaming examples:

$$Oldmovies1 := \pi_{movid,title}\ (\sigma_{year<1930}(Movie))$$

$$Oldmovies2[omid, omtitle] := \pi_{movid,title}\ (\sigma_{year<1930}(Movie))$$

on the fly renaming within an expression:

$$\ldots \bowtie \rho(Oldmovies, omid, omtitle)(\pi_{movid,title}\ (\sigma_{year<1930}(Movie)))$$

Examples

Library schema:

```
Book (bid, title, author)
Reader (rid, name, address, city)
Loan (bid, rid, ldate, rdate)
```

Q5: Give the names of the readers who borrowed at least two different Dickens-books

Relational algebra implementation

MonetDB: DBMS using MAL, a dialect of relational algebra

- developed at CWI, Amsterdam
- main-memory approach
- platform for analytical databases
- outperforms several commercial systems
- MAL is intermediate language for query processing
- SQL queries are translated to MAL and optimized
- *Pathfinder:* XQUERY queries are translated to MAL and optimized

Overview of algebra

Overview unary operators

- selection $\sigma_p(R)$ — p is selection predicate
- projection $\pi_L(R)$ — L is projection set
- renaming $\rho(R)$ — or using assignment

Overview binary operators

- union $R \cup S$ — schema compatibility
- difference $R - S$ — schema compatibility
- intersection $R \cap S$ — schema compatibility
- cartesian product $R \times S$
- theta-join $R \bowtie_\theta S$ — θ is matching condition
- natural join $R \bowtie S$
- division $R \div S$ — schema requirements