RELATIONAL ALGEBRA

Relational Querying

- Relational model helps in simple and powerful data retrieval
- Output of query modelled as a relation.
- Based on formal mathematical model.
  - *First order predicate Logic*
  - *Eg : Book('B101')*
- Allows for much optimization

Relational Data Languages

- Manipulation and Retrieval of data
- Two Types of Query Languages
  - *Relational Algebra*
    - Procedural
    - Set of operators operating on relations
  - *Relational Calculus*
    - NonProcedural
    - Users describe what they want rather than how to compute

Formal Relational Query Languages

- **Procedural** – (Relational Algebra)
  - *User specifies what data is required and how to get those data*
  - *Operational*
  - *Execution plans can be represented*
- **Nonprocedural** – **(Relational calculus)**
  - *User specifies what data is required without specifying how to get those data*
  - *Declarative*
  - *Query semantics can be represented*
- SQL is the most widely used query language based on Relational calculus.

Relational Algebra

- Procedural language
- Six basic operators
  - *select:* σ
  - *project:* Π
  - *union:* $\cup$
  - *set difference:* –
  - *Cartesian product:* x
  - *rename:* ρ
- The operators take one or two relations as inputs and produce a new relation as a result.

Relational Algebra

- Algebra ?
- Operands
  - Variables or values from which new values can be constructed.
- Operators
  - *Symbols denoting procedures to construct new values from Operands.*
- Relational Algebra has relations as operands and set operations as operators.
- Satisfies Closure property
  - *Output of an operation on relations is a relation itself*
  - *Operations can be composed.*

Types of Operations

- Unary Relational Operations
  - *Select (σ)*
  - *Project (Π)*
  - *Rename (ρ)*

- Binary Relational Operations
  - *Join* $\bowtie$
    - natural,semi,Θ-join
  - *Division (÷)*

- Set theory Operations
  - *Union (U)*
  - *Intersection (∩)*
  - *Difference (-)*
  - *Cartesian Product (X)*
- Additional Relational Operations
  - *Outer Joins*
  - *Outer Union*
  - *Aggregate Functions*
    - Eg. Sum,Count, Avg..

Select (denoted by σ (sigma))

- Retrieval of subset of the tuples from a relation based on a **selection condition**
- Selection condition acts as filter

$\sigma_{<selection\_condition>}$ (R) ; R is a relation

Selection condition is a boolean formula.

Tuples satisfying the condition are retained.

Ex : $\sigma_{ISBN='B110'}$ (Book)

Selection

- σ_F (R)

Class	Faculty
S7 CS	Dr. Albert
S7 CS	Dr. Neumann
S7 CS	Dr. Raj
S5 CS	Dr. Biswas

- Select * from class_tab where Class='S7CS'
- $\sigma_{Class='S7CS'}$ (class_tab)

Properties of Selection operation

- $\sigma_{<selection\_condition>}(R) = S$; R and S have same schema
- Number of tuples in S <= Number of tuples in R
- Is commutative

$$\sigma_{<cn1>}(\sigma_{<cn2>}(R)) = \sigma_{<cn2>}(\sigma_{<cn1>}(R))$$

- Cascade sequence of SELECT operations may be applied in any order:

$$\sigma_{<cn1>}(\sigma_{<cn2>}(\sigma_{<cn3>}(R))) = \sigma_{<cn2>}(\sigma_{<cn3>}(\sigma_{<cn1>}(R)))$$

- Cascade equivalent to conjunction of all the conditions

$$\sigma_{<cn1>}(\sigma_{<cn2>}(\sigma_{<cn3>}(R))) = \sigma_{<cn2>AND<cn3>AND<cn1>}(R)))$$

Project (denoted by Π (pi))

- Retrieval of the subset of columns from a relation based on a **specified list of attributes**
- Specified lists forms a projection of attributes

$\Pi_{<attr\_list>}(R)$; R is a relation

All the tuples of R with only the specified attribute values are retrieved.

Ex : $\Pi_{ISBN,\ Title}(Book)$

Is $\Pi_{ISBN}(\sigma_{Publ\_code='P010'}(Book))$ valid ?

Which property of relational algebra ?

BS <- $\sigma_{Publ\_code='P010'}$ (Book); BS2 <- Π_{ISBN} (BS))

Project Operation – Example

- Relation *r*:

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2

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

A	C
α	1
α	1
β	1
β	2

=

A	C
α	1
β	1
β	2

Projection

- $\Pi_{A,B}(R)$

COURSE	BOOK	FACULTY MEMBER
Distributed Databases	Pelaggatti	Dr. Albert
OS	Tanenbau m	Dr. Neumann
DBMS	Date	Dr. Susan
CN	Tanenbau m	Dr. Lekshmi
DBMS	Corth	Dr. Raj

- Select course, book from course_fac where course = ‘DBMS’
- $\Pi_{course,book}(\sigma_{Course=\text{‘DBMS’}}\ (course\_fac))$

COURSE	BOOK
DBMS	Date
DBMS	Corth

Properties of Projection operation

- $\Pi_{<attr\_list>}(R) = S;$
- Removes duplicate tuples. True ?
- Number of tuples in S <= Number of tuples in R

ISBN	Title	Category	Publ_code
B111	FISH	ARTICLE	P010
B112	GLOW	ARTICLE	P212
B110	FERT	NEWS	P010

$\Pi_{ISBN,\ Title}(Book)$

Attribute list contains Key.

$\Pi_{Publ\_code}(Book)$

Removes duplicate tuples.

Not commutative

$\Pi_{ISBN,\ Title}(\Pi_{category,publ\_code}(Book))$

$\Pi_{ISBN,\ Title}(\Pi_{category,ISBN}(Book))$

Results

ISBN	Title	Category	Publ_code
B111	FISH	ARTICLE	P010
B112	GLOW	ARTICLE	P212
B110	FERT	NEWS	P010

- $\Pi_{category,ISBN}(Book)$ $\Pi_{category}(Book)$

Category	ISBN
ARTICLE	B111
ARTICLE	B112
NEWS	B110

Category
ARTICLE
NEWS

- $\Pi_{ISBN,\ category}(\sigma_{Publ\_code='P010'}\ (Book))$

ISBN	Category	Publ_code
B111	ARTICLE	P010
B110	NEWS	P010

Rename (denoted by ρ (rho))

- The general RENAME operation ρ can be
  - $\rho_{S(B1,B2,\ldots,Bn)}(R)$ *changes*

  the relation name to S, *and* the column (attribute) names to B1, B1, …..Bn

- $\rho_{S}(R)$ changes:

  the *relation name only to S*

$\rho_{(B1,B2,\ldots,Bn)}(R)$ changes:

the *column (attribute) names only to B1, B1, …..Bn*

Union (denoted by U)

- RU S is a Binary Operation
- R and S should be type compatible
  - *R and S should have same number of attributes*
  - *Each pair of corresponding attributes must be type compatible (have same or compatible domains)*
- Tuples present in R or S or both are retrieved.
- Duplicate tuples are eliminated.
- Ex : Purchase_Invoice U Sales_Invoice

Union Operation – Example

- Relations *r, s:*

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

- $r \cup s$:

A	B
α	1
α	2
β	1
β	3

Union Operation

- Notation: $r \cup s$
- Defined as:

$$r \cup s = \{t \mid t \in r \text{ or } t \in s\}$$

- For $r \cup s$ to be valid.

1. *r, s* must have the *same* **arity** (same number of attributes)
2. The attribute domains must be **compatible** (example: 2nd column of *r* deals with the same type of values as does the 2nd column of *s*)

- Example: to find all courses taught in the Fall 2009 semester, or in the Spring 2010 semester, or in both

$$\Pi_{course\_id} (\sigma_{semester=\text{“Fall”} \ \wedge \ year=2009} (section)) \ \cup$$
$$\Pi_{course\_id} (\sigma_{semester=\text{“Spring”} \ \wedge \ year=2010} (section))$$

Union Example

- Find ISBN, title of the books that were published in 2009 or belongs to NEWS category

ISBN	Title	Year	Category	Publ_code
B111	FISH	2007	ARTICLE	P010
B112	GLOW	2009	ARTICLE	P212
B110	FERT	2010	NEWS	P010
B113	FINE ARTS	2009	NEWS	P010
B114	INDU – THE MAID	2008	NOVEL	P201

- Result1 <- $\Pi_{\text{ISBN, Title}}(\sigma_{\text{year=2009}}(\text{Book}))$
- Result2 <- $\Pi_{\text{ISBN, Title}}(\sigma_{\text{Category='NEWS'}}(\text{Book}))$
- Result <- Result1 U Result2

ISBN	Title	Year	Category	Publ_code
B112	GLOW	2009	ARTICLE	P212
B110	FERT	2010	NEWS	P010
B113	FINE ARTS	2009	NEWS	P010

Intersection

- Find ISBN, title of the books that were published in 2009 and belongs to NEWS category
- Result1 <- $\Pi_{\text{ISBN, Title}}(\sigma_{\text{year}=2009}\,(\text{Book}))$
- Result2 <- $\Pi_{\text{ISBN, Title}}(\sigma_{\text{Category}=\text{'NEWS'}}\,(\text{Book}))$
- Result <- Result1 ∩ Result2

ISBN	Title	Year	Category	Publ_code
B111	FISH	2007	ARTICLE	P010
B112	GLOW	2009	ARTICLE	P212
B110	FERT	2010	NEWS	P010
B113	FINE ARTS	2009	NEWS	P010
B114	INDU – THE MAID	2008	NOVEL	P201

ISBN	Title	Year	Category	Publ_code
B113	FINE ARTS	2009	NEWS	P010

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

Set Difference Operation

- Notation $r - s$
- Defined as:

$$r - s = \{t \mid t \in r \textbf{ and } t \notin s\}$$

- Set differences must be taken between **compatible** relations.
  - *r and s must have the same arity*
  - *attribute domains of r and s must be compatible*
- Example: to find all courses taught in the Fall 2009 semester, but not in the Spring 2010 semester

$$\Pi_{course\_id}\,(\sigma_{semester=\text{“Fall”}\ \land\ year=2009}\,(section))\ -$$
$$\Pi_{course\_id}\,(\sigma_{semester=\text{“Spring”}\ \land\ year=2010}\,(section))$$

Set difference of two relations

- Relations *r*, *s*:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

- *r* – *s:*

A	B
α	1
β	1

Set Difference

- Find ISBN, title of the books that were published in 2009 and does not belong to NEWS category
- Result1 <- $\Pi_{ISBN,\ Title}(\sigma_{year=2009}\ (Book))$
- Result2 <- $\Pi_{ISBN,\ Title}(\sigma_{Category=\text{'NEWS'}}\ (Book))$
- Result <- Result1 - Result2

ISBN	Title	Year	Category	Publ_code
B111	FISH	2007	ARTICLE	P010
B112	GLOW	2009	ARTICLE	P212
B110	FERT	2010	NEWS	P010
B113	FINE ARTS	2009	NEWS	P010
B114	INDU – THE MAID	2008	NOVEL	P201

ISBN	Title	Year	Category	Publ_code
B112	GLOW	2009	ARTICLE	P212

Set operations on different relations

P_Inv_No	Date	Publ_code
PI_1001	29/10/2009	P010
PI_2001	1/2/2001	P212
PI_1002	12/4/2007	P010
PI_1045	5/2/2006	P010

S_Inv_No	Date	Cust_code
SI_1001	29/10/2009	C010
SI_2001	1/2/2001	C212
SI_1002	12/4/2007	C010
SI_1045	5/2/2006	C010

Union

P_Inv_No	Date	Publ_code
PI_1001	29/10/2009	P010
PI_2001	1/2/2001	P212
PI_1002	12/4/2007	P010
PI_1045	5/2/2006	P010
SI_1001	29/10/2009	C010
SI_2001	1/2/2001	C212
SI_1002	12/4/2007	C010
SI_1045	5/2/2006	C010

Properties of Union, Intersect, Difference

- Commutative
  - *Satisfied by Union and Intersect*
- Associative
  - *Satisfied by Union and Intersect*
- Distributive
  - *R U (S-T) = (R U S) – (R U T)*
- R-(R-S) = Which Operation ?
- (R U S) – ((R – S) U (S- R)) = Which operation ?
- R-S ≠ S-R

Cartesian Product

- Combine tuples from two different relations
- Combinatorial manner
- *R X S*
- R(A1, A2, . . ., An) X S(B1, B2, . . ., Bm)
- Q(A1, A2, . . ., An, B1, B2, . . ., Bm) is the result
- Number of columns in Q

  cQ = cR + cS
- Number of tuples in Q

  *nQ = nR * nS*

Cartesian-Product Operation – Example

- Relations *r, s*:

r

A	B
α	1
β	2

s

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

- *r* x *s*:

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

CARTESIAN PRODUCT

ISBN	Title	Category	Publ_code
B111	FISH	ARTICLE	P010
B112	GLOW	ARTICLE	P212
B110	FERT	NEWS	P010

Publ_code	Name	Address
P011	Pub1	Add1
P212	Pub2	Add2
P010	Pub3	Add3

Book X Publisher

ISBN	Title	Category	Publ_code	Publ_code	Name	Address
B111	FISH	ARTICLE	P010	P011	Pub1	Add1
B111	FISH	ARTICLE	P010	P212	Pub2	Add2
B111	FISH	ARTICLE	P010	P010	Pub3	Add3
B112	GLOW	ARTICLE	P212	P011	Pub1	Add1
B112	GLOW	ARTICLE	P212	P212	Pub2	Add2
B112	GLOW	ARTICLE	P212	P010	Pub3	Add3
......				.....		

Composition of Operations

- Can build expressions using multiple operations
- Example: $\sigma_{A=C}(r \times s)$

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

- *r x s*

A	B	C	D	E
α	1	α	10	a
β	2	β	10	a
β	2	β	20	b

- $\sigma_{A=C}(r \times s)$

Joins

- To give meaningful representation for the cartesian product.

ISBN	Title	Category	Publ_code
B111	FISH	ARTICLE	P010
B112	GLOW	ARTICLE	P212
B110	FERT	NEWS	P010

Publ_code	Name	Address
P011	Pub1	Add1
P212	Pub2	Add2
P010	Pub3	Add3

Book ⋈ Publisher

ISBN	Title	Category	Publ_code
B111	FISH	ARTICLE	P010
B112	GLOW	ARTICLE	P212
B110	FERT	NEWS	P010

Publ_code	Name	Address
P010	Pub3	Add3
P212	Pub2	Add2
P010	Pub3	Add3

Joined Relations

- **Join operations** take two relations and return as a result another relation.
- A join operation is a Cartesian product which requires that tuples in the two relations match (under some condition). It also specifies the attributes that are present in the result of the join
- The join operations are typically used as subquery expressions in the **from** clause

Joined Relations

- **Join operations** take two relations and return as a result another relation.
- These additional operations are typically used as subquery expressions in the **from** clause
- **Join condition** – defines which tuples in the two relations match, and what attributes are present in the result of the join.
- **Join type** – defines how tuples in each relation that do not match any tuple in the other relation (based on the join condition) are treated.

Join types
inner join
left outer join
right outer join
full outer join

Join Conditions
natural
on <predicate>
using $(A_1, A_1, \ldots, A_n)$

Join (denoted by ⋈)

- Derivative of Cartesian product
- Allows to combine tuples from different relations based on some meaningful condition
- Θ-join
  - *Join based on any of the binary comparison operators (>,=,<,>=,>=,!= et. al)*
  - *Any Boolean formula*

- $R \bowtie_{<F>} S$; F is a join condition
- $F = R.a\ \Theta\ S.b$
- Can you Express $R \bowtie_F S$ in terms of other operation ?

Join Example

- Get the publishers name of each book

ISBN	Title	Category	Pub_cd
B111	FISH	ARTICLE	P010
B112	GLOW	ARTICLE	P212
B110	FERT	NEWS	P010

Pbl_code	Publ_name	Publ_phone
P212	Pearson	3452198
P010	McGraw	8930287

– $Book \bowtie_{pub\_cd=pbl\_code} Publisher$

Natural Join

- Get the publishers name of each book

ISBN	Title	Category	Publ_code
B111	FISH	ARTICLE	P010
B112	GLOW	ARTICLE	P212
B110	FERT	NEWS	P010

Publ_code	Publ_name	Publ_phone
P212	Pearson	3452198
P010	McGraw	8930287

  - *Book * Publisher*
  - *The join condition is dependent on the columns with same attribute names*

Natural Join 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	ε

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

- Find the names of all instructors in the Comp. Sci. department together with the course titles of all the courses that the instructors teach
  - $\Pi_{name,\ title}\ (\sigma_{dept\_name=\text{"Comp. Sci."}}\ (instructor \bowtie teaches \bowtie course))$
- Natural join is associative
  - $(instructor \bowtie teaches) \bowtie course$ *is equivalent to* $instructor \bowtie (teaches \bowtie course)$
- Natural join is commutative
  - $instruct \bowtie teaches$ *is equivalent to* $teaches \bowtie instructor$

Joins - Example

- Relation *course*

course_id	*title*	*dept_name*	*credits*
BIO-301	Genetics	Biology	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3

- Relation *prereq*

course_id	*prereq_id*
BIO-301	BIO-101
CS-190	CS-101
CS-347	CS-101

- Observe that

  prereq information is missing for CS-315 and

  course information is missing for CS-437

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.*

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

- Inner 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

Null Values

- It is possible for tuples to have a null value, denoted by *null*, for some of their attributes
- *null* signifies an unknown value or that a value does not exist.
- The result of any arithmetic expression involving *null* is *null.*
- Aggregate functions simply ignore null values
- For duplicate elimination and grouping, null is treated like any other value, and two nulls are assumed to be the same

Division(Quotient)

- Select those rows which are sufficient to provide all values in certain columns
  - *Which publishers publishes all types of books ?*
  - *R = Book ; S = Category*
  - $R \div S = \{(x) \mid \exists (x,y) \in R, \forall y \in S\}$
  - $= \Pi_A(R) - \Pi_A((\Pi_A(R) X\ S) - R)$
  - *A : set of attributes not in S*
  - *Let r and s be relations on schemas R and S respectively where*
  - $R = (A1, ..., Am, B1, ..., Bn)$
  - $S = (B1, ..., Bn)$
  - *Then r/s is a relation on schema* $R - S = (A1, ..., Am)$

ISBN	Title	Category	Publ_code
B111	FISH	ARTICLE	P010
B112	GLOW	ARTICLE	P212
B110	FERT	NEWS	P010

Category
ARTICLE
NEWS

Example of Division A/B

A

sno	pno
s1	p1
s1	p2
s1	p3
s1	p4
s2	p1
s2	p2
s3	p2
s4	p2
s4	p4

B1

pno
p2

B2

pno
p2
p4

B3

pno
p1
p2
p4

A/B1

sno
s1
s2
s3
s4

A/B2

sno
s1
s4

A/B3

sno
s1

Division 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

$r \div s$:

A	B	C
α	a	γ
γ	a	γ

Questions

- Find names of Publishers who have published book B101

- Find the names of sales representatives who work for the publishers who have published books in News as well as in Article category

Aggregate functions

- Max
- Min
- Sum
- Count
- Average

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 Operation – Example

- Relation *r*:

A	*B*	*C*
α	α	7
α	β	7
β	β	3
β	β	10

$\mathcal{g}_{\mathbf{sum(c)}}(\mathrm{r})$

sum-C
27

Aggregate Operation – 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

$$_{branch\text{-}name}\,\mathcal{G}_{\,sum(balance)}\,(account)$$

branch-name	*balance*
Perryridge	1300
Brighton	1500
Redwood	700

Aggregate Functions (Cont.)

- 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*

$$_{\textit{branch-name}}\,\mathcal{G}\,_{\textbf{sum}(\textit{balance})\ \textbf{as}\ \textit{sum-balance}}(\textit{account})$$

Advantages of Relational Algebra

- Rigorously defined simple and yet powerful query language
- More operational
- Useful for query evaluation plans
- Several ways of expressing a query. Query optimizer should choose the most efficient one.

THANK YOU

References

- Silberschatz A Korth H F and SudharshanS , “Database System Concepts”, 6th Edition, TMH publishing company limited, 2011.