Relational Algebra

Lecture 4A

Kathleen Durant
Northeastern University

Relational Query Languages

- Query languages: Allow manipulation and retrieval of data from a database.
- Relational model supports simple, powerful QLs:
  - Strong formal foundation based on logic.
  - Allows for optimization.
  - Query Languages != programming languages
  - QLs not expected to be “Turing complete”.
  - QLs not intended to be used for complex calculations.
  - QLs support easy, efficient access to large data sets.

Relational Query Languages

- Two mathematical Query Languages form the basis for "real" query languages (e.g. SQL), and for implementation:
- Relational Algebra: More operational, very useful for representing execution plans.
  - Basis for SEQUEL
- Relational Calculus: Let's users describe WHAT they want, rather than HOW to compute it. (Non-operational, declarative.)
  - Basis for QUEL

Mathematical Foundations: Cartesian Product

- Let:
  - A be the set of values { a_1, a_2, … }
  - B be the set of values { b_1, b_2, … }
  - C be the set of values { c_1, c_2, … }

- The Cartesian product of A and B (written A x B) is the set of all possible ordered pairs (a_i, b_j), where $a_i \in A$ and $b_j \in B$.
- Similarly:
  - A x B x C is the set of all possible ordered triples (a_i, b_j, c_k), where $a_i \in A$, $b_j \in B$, and $c_k \in C$.
  - A_1 x A_2 x … x A_n is the set of all possible ordered *tuples* (a_{1i}, a_{2j}, …, a_{nk}), where $a_{de} \in A_d$

Cartesian Product Example

- A = {small, medium, large}
- B = {shirt, pants}

A X B	Shirt	Pants
Small	(Small, Shirt)	(Small, Pants)
Medium	(Medium, Shirt)	(Medium, Pants)
Large	(Large, Shirt)	(Large, Pants)

- A x B = {(small, shirt), (small, pants), (medium, shirt), (medium, pants), (large, shirt), (large, pants)}
  - Set notation

Example: Cartesian Product

- What is the Cartesian Product of AxB ?
  - A = {perl, ruby, java}
  - B = {necklace, ring, bracelet}

- What is BxA?

A x B	Necklace	Ring	Bracelet
Perl	(Perl,Necklace)	(Perl, Ring)	(Perl,Bracelet)
Ruby	(Ruby, Necklace)	(Ruby,Ring)	(Ruby,Bracelet)
Java	(Java, Necklace)	(Java, Ring)	(Java, Bracelet)

Mathematical Foundations: Relations

- The domain of a variable is the set of its possible values
- A relation on a set of variables is a subset of the Cartesian product of the domains of the variables.
  - Example: let x and y be variables that both have the set of non-negative integers as their domain
  - {(2,5),(3,10),(13,2),(6,10)} is one relation on (x, y)
- A table is a subset of the Cartesian product of the domains of the attributes. Thus a **table is a mathematical relation**.
- Synonyms:
  - Table = relation
  - Row (record) = tuple
  - Column (field) = attribute

Mathematical Relations

- In tables, as, in mathematical relations, the order of the tuples does not matter but the order of the attributes does.
- The domain of an attribute usually includes NULL, which indicates the value of the attribute is unknown.

What is an Algebra?

- Mathematical system consisting of:

- Operands --- variables or values from which new values can be constructed.

- Operators --- symbols denoting procedures that construct new values from given values.

What is Relational Algebra?

- An algebra whose operands are relations or variables that represent relations.
- Operators are designed to do the most common things that we need to do with relations in a database.
- The result is an algebra that can be used as a query language for relations.

Relational Algebra

- A collection of operations that users can perform on relations to obtain a desired result
- This is an introduction and only covers the algebra needed to represent SQL queries
  - Select, project, rename
  - Cartesian product
  - Joins (natural, condition, outer)
  - Set operations (union, intersection, difference)
- Relational Algebra treats relations as sets: duplicates are removed

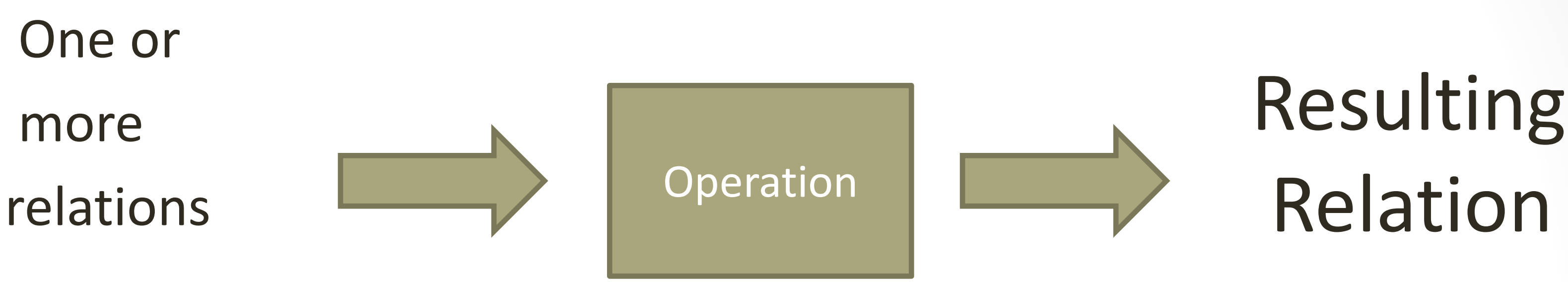
Relation Instance vs. Schema

- Schema of a relation consists of
  - The name of the relation
  - The fields of the relation
  - The types of the fields
- For the Student table

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32

- Schema = Student(SID int, Name char(20), Login char(20), DoB date, GPA real)
- Instance of a relation is an actual collection of tuples
  - Table with rows of values
- Database schema is the schema of the relations in a database

Relational Algebra



For each operation:
both the operands and the result are relations

Facts on relational algebra queries

- A query is applied to relation instances, and the result of a query is also a relation instance.
  - Schemas of input relations for a query are fixed
  - But query will run regardless of instance.
- The schema for the **result** of a given query is also fixed
  - Determined by definition of query language constructs.
- Positional vs. named-field notation:
  - Positional notation easier for formal definitions, named field notation more readable.
  - Both used in SQL

Basic Relational Algebra Operations

- Basic operations:
- Selection (σ): Selects a subset of tuples from a relation.
- Projection (π): Selects columns from a relation.
- Cross-product ($\times$): Allows us to combine two relations.
- Set-difference ($-$): Tuples in relation 1, but not in relation 2.
- Union ($\cup$): Tuples in relation 1 and in relation 2.
- Additional operations:
  - Intersection, join, division, renaming: Not essential, but (very) useful.
- Each operation returns a relation, operations can be composed (Algebra is "closed")
  - Contains the closure property
- Since operators' input is a relation and its output is a relation we can string these operators together to form a more complex operator

Basic Operation: Projection π

- Deletes attributes that are not in projection list.
- Schema of result contains exactly the fields in the projection list, with the same names that they had in the input relation.
- Syntax: $\pi_{f1, f2, \ldots}$ (Relation)
- Projection operator has to eliminate duplicates. (Why?)
  - Note: real systems typically do not eliminate duplicates unless the user explicitly asks for it. (Why not?)

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32

$\pi_{Sid, Name}(S1)$

SID	Name
55515	Smith
55516	Jones
55517	Ali
55518	Smith

$\pi_{Sid}(S1)$

SID
55515
55516
55517
55518

Basic Operations: Select σ

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32

- Selects rows that satisfy the selection condition.
  - No duplicates in result (Why?)
  - Schema of result is identical to schema of input relation
- Selection predicates can include: <, >, =, !=, and, or, not
  - Examples:
  - $\sigma_{Sid\ !=\ 55516}(S1)$
  - $\sigma_{Name\ =\ 'Smith'}(S1)$

$$\sigma_{Sid\ >\ 55516}(S1)$$

SID	Name	Login	DoB	GPA
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32

- Syntax: $\sigma_{Conditional}(\text{Relation})$

Operator composition example.

Select and Project

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32

$$\pi_{Sid, Name}(\sigma_{Sid > 55516}(S1))$$

SID	Name
55517	Ali
55518	Smith

Union ∪

- Takes two input relations, which must be union-compatible:
  - Same number of fields.
  - `Corresponding' fields have the same type.

S1

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32

S2

SID	Name	Login	DoB	GPA
55515	Chen	chen@ccs	Jan 10,1990	3.01
55519	Alton	alton@hist	Jun 11, 1992	2.07
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32

S1 U S2

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32
55515	Chen	chen@ccs	Jan 10,1990	3.01
55519	Alton	alton@hist	Jun 11, 1992	2.07

Intersection ∩

Occurs in S1 and S2

S1 ∩ S2

SID	Name	Login	DoB	GPA
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32

Set difference

Occurs in S1 but not in S2

S1 – S2

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98

S1

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32

S2

SID	Name	Login	DoB	GPA
55515	Chen	chen@ccs	Jan 10,1990	3.01
55519	Alton	alton@hist	Jun 11, 1992	2.07
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32

Cartesian Product

- Also referred to as *cross-product* or *product*.
- Two input relations.
- Each tuple of the one relation is paired with each tuple of the other relation.
- *Result schema* has one attribute per attribute of both input relations, with attribute names `inherited’ if possible.
- In the result, there may be two attributes with the same name, e.g. both S1 and R1 have an attribute called *sid*.
- Then, apply the *renaming operation*, e.g.

$$\rho_{S1(1 \to sid1, 5 \to sid2)}(R1)$$

Cross-Product x

S1

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98

Each row within S1 is paired with each row of C1

C1

CId	Grade
History 101	C
Biology 220	A
Anthro 320	B
Music 101	A

S1 x C1

SID	Name	Login	DoB	GPA	CID	Grade
55515	Smith	smith@ccs	Jan 10,1990	3.82	History 101	C
55515	Smith	smith@ccs	Jan 10,1990	3.82	Biology 220	A
55515	Smith	smith@ccs	Jan 10,1990	3.82	Anthro 320	B
55515	Smith	smith@ccs	Jan 10,1990	3.82	Music 101	A
55516	Jones	jones@hist	Feb 11, 1992	2.98	History 101	C
55516	Jones	jones@hist	Feb 11, 1992	2.98	Biology 220	A
55516	Jones	jones@hist	Feb 11, 1992	2.98	Anthro 320	B
55516	Jones	jones@hist	Feb 11, 1992	2.98	Music 101	A

Result schema has one field per field of S1 and C1, with field names `inherited' if possible.

Rename fields or table

- Renames relations / attributes, without changing the relation instance.
- $\rho_{S(A1, A2, \ldots, An)}(R)$

  relation R is renamed to S,
  attributes are renamed A1, . . ., An
- Rename only some attributes

  $\rho_{S(1->A1, \ldots, k->Ak)}(R)$

  using the positional notation to reference attributes
- No renaming of attributes

  $\rho_{S}(R)$

Rename ρ

- Reassign the field names

S1

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98

ρ(C(1-> S1.sid, 6->C1.sid), S1 X C1)

C1

Sid	CId	Grad e
55515	History 101	C
55516	Biology 220	A
55517	Anthro 320	B
55518	Music 101	A

C

S1.SID	Name	Login	DoB	GPA	C1.Sid	CID	Grade
55515	Smith	smith@ccs	Jan 10,1990	3.82	55515	History 101	C
55515	Smith	smith@ccs	Jan 10,1990	3.82	55516	Biology 220	A
55515	Smith	smith@ccs	Jan 10,1990	3.82	55517	Anthro 320	B
55515	Smith	smith@ccs	Jan 10,1990	3.82	55518	Music 101	A
55516	Jones	jones@hist	Feb 11, 1992	2.98	55515	History 101	C
55516	Jones	jones@hist	Feb 11, 1992	2.98	55516	Biology 220	A
55516	Jones	jones@hist	Feb 11, 1992	2.98	55517	Anthro 320	B
55516	Jones	jones@hist	Feb 11, 1992	2.98	55518	Music 101	A

Prepend the name of the original relation to the fields having a collision

Naming columns and result set to C

Conditional Join

S1

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98

- Accepts a conditional
- Operation equivalent to:
- $S1 \bowtie_C C1 = \sigma_C (S1 \times C1))$
- Filters out tuples according to the conditional expression

C1

CId	Grade
History 101	C
Biology 220	A
Anthro 320	B
Music 101	A

- **$S1 \bowtie_{gpa > 3.0} C1$**

SID	Name	Login	DoB	GPA	CID	Grade
55515	Smith	smith@ccs	Jan 10,1990	3.82	History 101	C
55515	Smith	smith@ccs	Jan 10,1990	3.82	Biology 220	A
55515	Smith	smith@ccs	Jan 10,1990	3.82	Anthro 320	B
55515	Smith	smith@ccs	Jan 10,1990	3.82	Music 101	A

Conditional Join is equivalent to:
Cartesian project (x)
Selection (σ)

Equijoin $\bowtie_C$

S1

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98

- What does it do: performs a filtered Cartesian product
- Filters out tuples where the attribute that have the same name have a different value
- $\mathbf{S} \bowtie_{\mathbf{S1.sid = C1.sid}} \mathbf{C1}$

C1

Sid	CId	Grad e
55515	History 101	C
55516	Biology 220	A
55517	Anthro 320	B
55518	Music 101	A

SID	Name	Login	DoB	GPA	CID	Grade
55515	Smith	smith@ccs	Jan 10,1990	3.82	History 101	C
55516	Jones	jones@hist	Feb 11, 1992	2.98	Biology 220	A

Only one copy of Sid is in the resultant relation

Natural Join

S1

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98

- What does it do: performs a filtered Cartesian product
- Filters out tuples where the attribute that have the same name have a different value
- **S1 $\bowtie$ C1**

No need to specify field list

C1

Sid	CId	Grade
55515	History 101	C
55516	Biology 220	A
55517	Anthro 320	B
55518	Music 101	A

SID	Name	Login	DoB	GPA	CID	Grade
55515	Smith	smith@ccs	Jan 10,1990	3.82	History 101	C
55516	Jones	jones@hist	Feb 11, 1992	2.98	Biology 220	A

Natural join is equivalent to:
Cartesian project (x)
Selection (σ)
Projection (π)

Precedence of Relational Operators

- 1. [σ, π, ρ] (highest).
- 2. [X, ⋈]
- 3. ∩
- 4. [∪, —]

Schema of the Resulting Table

- Union, intersection, and difference operators
  - the schemas of the two operands must be the same, so use that schema for the result.
- Selection operator
  - schema of the result is the same as the schema of the operand.
- Projection operator
  - list of attributes determines the schema.

Relational Algebra: Summary

- The relational model has rigorously defined query languages that are simple and powerful.
  - Relational algebra is more operational
  - Useful as internal representation for query evaluation plans
- Several ways of expressing a given query
- A query optimizer should choose the most efficient version

Relational Algebra

- Like ERM modeling there are many ways to solve the problem at hand
- Given the theory behind RA, a sophisticated query optimization engineer can write algorithms that optimize a query
  - Theory in practice