Relational Algebra

Instructor: Lei Jiang

Slides provided by Ramona Truta

Outline

- How to ask queries in the Relational Model.
- Relational Algebra Operators
  - Set-theory Operators: Union, Intersection, Difference
  - Selection and Projection
  - Renaming
  - Cartesian Product
  - Natural Join
  - Division
- Relational Algebra Query Examples

Describing data: what is a data model?

- Mathematical representation of data.

PLUS

- Constraints.
- Operations.

Examples of queries

- Find titles of movies.
- Find artists who played in movies directed by Polanski.
- Find the artists who played the highest number of roles in a movie.
- ...

What is the result of a query?

- They are relations constructed from the relations in the database.

How to ask a query?

- Query languages
  - Commercial: SQL
  - Theoretical: Relational calculus, algebra, datalog etc

Query Languages: Declarative vs Procedural

- In our queries, we ask
  - **what** we want to see in the output,
  - but we do not say **how** we want to get this output.
- **Declarative** query languages: they specify what is needed in the output, but do not say how to get it.
- **Procedural** languages: specify how to get the result.
  - DBMS figures out **how** to get the result, and gives it to the user.

Query Languages: Theoretical vs Practical

- **Theoretical languages**:
  - Declarative: relational calculus, rule-based queries
  - Procedural: relational algebra
- **Practical languages**: mix of both, but mostly one uses declarative features.

Procedural Language: Relational Algebra

- A collection of algebraic operators that
  - Are defined on relations;
  - produce a relation as result;
  - and, therefore, can be combined to form complex algebraic operations.
- Operators:
  - Set operators: Union ($\cup$), Intersection ($\cap$), Difference ($-$);
  - Renaming ρ;
  - Selection (σ) and Projection (π);
  - Join: Natural Join ($\bowtie$), Cartesian Product ($\times$), Theta Join ($\bowtie_\theta$)
  - Division ($\div$)

Set Operators: Union, Intersection, Difference

- Relations are sets, so we can apply set-theoretic operators.
- However, we want the results to be relations (that is, homogeneous sets of tuples).
- It is therefore meaningfull to only apply $\cup, \cap, -$ to pairs of relations defined over the same attributes.

Union

- `Graduates`:

Number	Surname	Age
7255	Kubrick	25
7342	Altman	28
7111	Polanski	30

- `Managers`:

Number	Surname	Age
7155	Robinson	45
7111	Polanski	30

- `Graduates` $\cup$ `Managers`:

Union cont'd

- `Graduates`:

Number	Surname	Age
7255	Kubrick	25
7342	Altman	28
7111	Polanski	30

- `Managers`:

Number	Surname	Age
7155	Robinson	45
7111	Polanski	30

- `Graduates` $\cup$ `Managers`:

Number	Surname	Age
7255	Kubrick	25
7342	Altman	28
7111	Polanski	30
7155	Robinson	45

Intersection

- `Graduates`:

Number	Surname	Age
7255	Kubrick	25
7342	Altman	28
7111	Polanski	30

- `Managers`:

Number	Surname	Age
7155	Robinson	45
7111	Polanski	30

- `Graduates` $\cap$ `Managers`:

Intersection cont'd

- `Graduates`:

Number	Surname	Age
7255	Kubrick	25
7342	Altman	28
7111	Polanski	30

- `Managers`:

Number	Surname	Age
7155	Robinson	45
7111	Polanski	30

- `Graduates` $\cap$ `Managers`:

Number	Surname	Age
7111	Polanski	30

Difference

- `Graduates`:

Number	Surname	Age
7255	Kubrick	25
7342	Altman	28
7111	Polanski	30

- `Managers`:

Number	Surname	Age
7155	Robinson	45
7111	Polanski	30

- `Graduates` − `Managers`:

- `Managers` − `Graduates`:

Difference cont'd

- `Graduates`:

Number	Surname	Age
7255	Kubrick	25
7342	Altman	28
7111	Polanski	30

- `Managers`:

Number	Surname	Age
7155	Robinson	45
7111	Polanski	30

- `Graduates` − `Managers`:

Number	Surname	Age
7255	Kubrick	25
7342	Altman	28

- `Managers` − `Graduates`:

Number	Surname	Age
7155	Robinson	45

A meaningfull but impossible union

- `Paternity`:

Father	Child
George	Elizabeth
Philip	Charles
Charles	William

- `Maternity`:

Mother	Child
Elizabeth	Charles
Elizabeth	Andrew

- We want their union, which should be the "parent-child" relation.

  – But we cannot use `Maternity` $\cup$ `Paternity` because they have different attributes!

- Solution: *rename* the attributes!

Renaming

- This is a unary operator which changes the attribute names for a relation without changing any values.
- Renaming eliminates the limitations associated with set operators.
- Notation: $\rho_{\texttt{new\_name} \leftarrow \texttt{old\_name}}(\texttt{R})$
- Then, $\rho_{\texttt{Parent} \leftarrow \texttt{Father}}(\texttt{Paternity}) \cup \rho_{\texttt{Parent} \leftarrow \texttt{Mother}}(\texttt{Maternity})$:

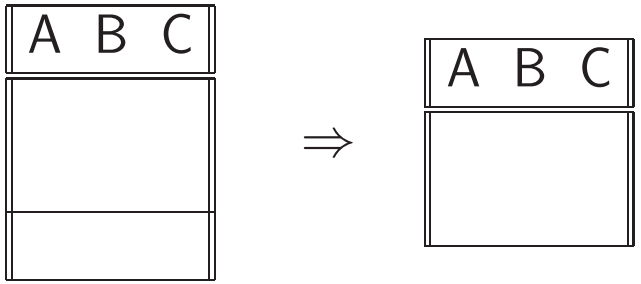
Renaming cont'd

- This is a unary operator which changes the attribute names for a relation without changing any values.
- Renaming eliminates the limitations associated with set operators.
- Notation: $\rho_{\texttt{new\_name} \leftarrow \texttt{old\_name}}(\texttt{R})$
- Then, $\rho_{\texttt{Parent} \leftarrow \texttt{Father}}(\texttt{Paternity}) \cup \rho_{\texttt{Parent} \leftarrow \texttt{Mother}}(\texttt{Maternity})$:

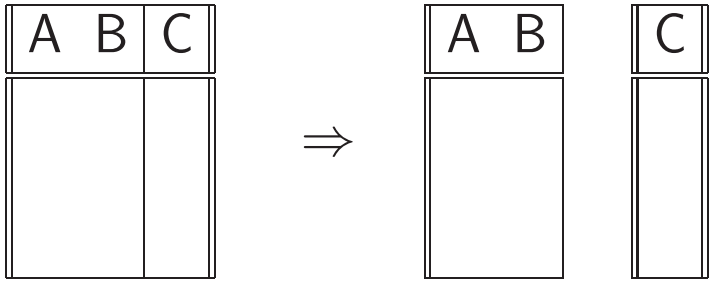
Parent	Child
George	Elizabeth
Philip	Charles
Charles	William
Elizabeth	Charles
Elizabeth	Andrew

Selection and Projection

- They are unary operators.
- Selection for "horizontal" decompositions;



- Projection for "vertical" decompositions.



Selection

- This is a unary operator which returns a relation
  - with the same schema as the operand,
  - but only with the tuples satisfying a condition.
- Notation: $\sigma_{cond}(R)$
- Semantics: $\sigma_{\texttt{cond}}(\texttt{R}) = \{\texttt{t} | \texttt{t} \in \texttt{R st cond(t)}\}$
  - `cond` is a formula in propositional calculus consisting of `terms` connected by $\wedge, \vee, \neg$
  - Each `term` is one of:
    - `<attribute> op <attribute>` or
    - `<attribute> op <constant>`
    - where `op` is one of $=, \neq, >, \geq, <, \leq$

Selection: examples

- `Graduates`:

Number	Surname	Age
7255	Kubrick	25
7342	Altman	28
7111	Polanski	30

- $\sigma_{\texttt{Number}>7000}(\texttt{Graduates})$

Selection: examples cont'd

- `Graduates`:

Number	Surname	Age
7255	Kubrick	25
7342	Altman	28
7111	Polanski	30

- $\sigma_{\texttt{Number}>7000}(\texttt{Graduates})$

Number	Surname	Age
7255	Kubrick	25
7342	Altman	28
7111	Polanski	30

Selection: examples cont'd

- `Graduates`:

Number	Surname	Age
7255	Kubrick	25
7342	Altman	28
7111	Polanski	30

- $\sigma_{\texttt{Number}<7000}(\texttt{Graduates})$

Number	Surname	Age

- $\sigma_{\texttt{Surname}\neq\texttt{'Kubrick'}\wedge\texttt{Age}\geq 30}(\texttt{Graduates})$

Selection: examples cont'd

- `Graduates`:

Number	Surname	Age
7255	Kubrick	25
7342	Altman	28
7111	Polanski	30

- $\sigma_{\texttt{Number}<7000}(\texttt{Graduates})$

Number	Surname	Age

- $\sigma_{\texttt{Surname}\neq\texttt{'Kubrick'}\wedge\texttt{Age}\geq 30}(\texttt{Graduates})$

Number	Surname	Age
7111	Polanski	30

Projection

- This is a unary operator which returns a relation
  - which includes all the tuples
  - but only a subset of the attributes of the operand.
- Notation: Given $\mathtt{R}(\mathtt{X}), \mathtt{Y} \subseteq \mathtt{X}$: $\pi_Y(R)$
  - only leaves attributes $\mathtt{Y}$ in relation $\mathtt{R}$.
- Semantics: $\pi_{\mathtt{Y}}(\mathtt{R}) = \{\mathtt{t}[\mathtt{Y}] | \mathtt{t} \in \mathtt{R}\}$

Projection: examples

- `Graduates`:

Number	Surname	Age
7255	Kubrick	25
7111	Polanski	30
7211	Polanski	30

- $\pi_{\texttt{Number,Age}}(\texttt{Graduates})$

- $\pi_{\texttt{Surname}}(\texttt{Graduates})$

Projection: examples cont'd

- `Graduates`:

Number	Surname	Age
7255	Kubrick	25
7111	Polanski	30
7211	Polanski	30

- $\pi_{\texttt{Number,Age}}(\texttt{Graduates})$

Number	Age
7255	25
7111	30
7211	30

- $\pi_{\texttt{Surname}}(\texttt{Graduates})$

Surname
Kubrick
Polanski

Cartesian Product

- Puts together two relations
- Notation: $\mathtt{r} \times \mathtt{s}$
- Semantics: $\mathtt{r} \times \mathtt{s} = \{\mathtt{tq} \mid \mathtt{t} \in \mathtt{r} \text{ and } \mathtt{q} \in \mathtt{s}\}$
  - puts together each tuple t of r with each tuple q of s
- Assume that the attributes of r and s are disjoint;
  - if they are not disjoint, in the result they have to be renamed.
- If r has n tuples and s has m tuples, then $r \times s$ has $n \times m$ tuples

Cartesian Product: example

- r:

Employee	Department
Smith	Sales
Blum	Production
Lewis	Production

s:

Department	Head
Sales	Mori
Production	Brown

- `r` $\times$ `s`:

Cartesian Product: example

- r:

Employee	Department
Smith	Sales
Blum	Production
Lewis	Production

s:

Department	Head
Sales	Mori
Production	Brown

- `r` $\times$ `s`:

Employee	Department	Department1	Head
Smith	Sales	Sales	Mori
Smith	Sales	Production	Brown
Blum	Production	Sales	Mori
Blum	Production	Production	Brown
Lewis	Production	Sales	Mori
Lewis	Production	Production	Brown

Join

- The most used operator in Relational Algebra.
- Allows us to connect data among different relations.
- Two main versions of the join:
  - Natural join: $\bowtie$
  - Theta join: $\mathtt{r} \bowtie_\theta \mathtt{s} = \sigma_\theta(\mathtt{r} \times \mathtt{s})$

Natural Join: example

- r:

Employee	Department
Smith	Sales
Blum	Production
Lewis	Production

- s:

Department	Head
Sales	Mori
Production	Brown

- We want a new relation, `t(Employee, Department, Head)`, such that `Head` is the head of the `Department` where `Employee` works.
- One possible solution:
  - $r \times s$
  - Is it really?

Natural Join: a better solution

- r:

Employee	Department
Smith	Sales
Blum	Production
Lewis	Production
Kifer	HR

s:

Department	Head
Sales	Mori
Production	Brown

- $\mathtt{t} = \mathtt{r} \bowtie \mathtt{s}$:

Natural Join: a better solution cont'd

- r:

Employee	Department
Smith	Sales
Blum	Production
Lewis	Production
Kifer	HR

s:

Department	Head
Sales	Mori
Production	Brown

- $\mathtt{t} = \mathtt{r} \bowtie \mathtt{s}$:

Employee	Department	Head
Smith	Sales	Mori
Blum	Production	Brown
Lewis	Production	Brown

(Optional) Natural Join: it's not a new operator!

- $\mathtt{r} \times \mathtt{s}$:

Employee (r.Employee)	Department (r.Department)	Department1 (s.Department)	Head (s.Head)
Smith	Sales	Sales	Mori
Smith	Sales	Production	Brown
Blum	Production	Sales	Mori
Blum	Production	Production	Brown
Lewis	Production	Sales	Mori
Lewis	Production	Production	Brown

- $\mathtt{r} \bowtie \mathtt{s} = \pi_{\mathtt{Employee,Department,Head}}(\sigma_{\mathtt{r.Department=s.Department}}(\mathtt{r} \times \mathtt{s}))$

  – The tuples in the result are obtained by combining tuples from the operands that have equal values on the common attributes.

Natural Join: another example

- r:

Employee	Department	Company
Smith	Sales	Google
Blum	Production	Microsoft
Lewis	Production	Oracle
Kifer	HR	Pearson

- s:

Department	Head	Company
Sales	Mori	Google
Production	Brown	Oracle

- $\mathtt{t} = \mathtt{r} \bowtie \mathtt{s}$:

Natural Join: another example cont'd

- r:

Employee	Department	Company
Smith	Sales	Google
Blum	Production	Microsoft
Lewis	Production	Oracle
Kifer	HR	Pearson

- s:

Department	Head	Company
Sales	Mori	Google
Production	Brown	Oracle

- $\mathtt{t} = \mathtt{r} \bowtie \mathtt{s}$:

Employee	Department	Head	Company
Smith	Sales	Mori	Google
Lewis	Production	Brown	Oracle

Natural Join: the general case

- $\mathtt{r}(\mathtt{A}_1, \mathtt{A}_2, \cdots, \mathtt{A}_n, \mathtt{B}_1, \mathtt{B}_2, \cdots, \mathtt{B}_m), \mathtt{s}(\mathtt{A}_1, \mathtt{A}_2, \cdots, \mathtt{A}_n, \mathtt{C}_1, \mathtt{C}_2, \cdots, \mathtt{C}_p)$
- $\mathtt{r} \bowtie \mathtt{s}$ is a relation on $\mathtt{A}_1, \mathtt{A}_2, \cdots, \mathtt{A}_n, \mathtt{B}_1, \mathtt{B}_2, \cdots, \mathtt{B}_m, \mathtt{C}_1, \mathtt{C}_2, \cdots, \mathtt{C}_p$:

$$\pi_{\mathtt{A}_1,\ldots,\mathtt{A}_n,\ \mathtt{B}_1,\ldots,\mathtt{B}_k,\mathtt{C}_1,\ldots,\mathtt{C}_m}(\sigma_{\mathtt{r.A}_1=\mathtt{s.A}_1 \wedge \ldots \wedge \mathtt{r.A}_n=\mathtt{s.A}_n}(\mathtt{r} \times \mathtt{s}))$$

Natural Join: definition and properties

- **Definition**:
  - $\mathtt{r}(\mathtt{X}_1), \mathtt{s}(\mathtt{X}_2)$
  - $\mathtt{r} \bowtie \mathtt{s}$ is a relation on $\mathtt{X}_1\mathtt{X}_2$ (the union of the two sets):
    - $\{\mathtt{t} \text{ on } \mathtt{X}_1\mathtt{X}_2 \mid \mathtt{t}[\mathtt{X}_1] \in \mathtt{r} \wedge \mathtt{t}[\mathtt{X}_2] \in \mathtt{s}\}$

    or, equivalently:
    - $\{\mathtt{t} \text{ on } \mathtt{X}_1\mathtt{X}_2 \mid \exists \mathtt{t}_1 \in \mathtt{r} \wedge \exists \mathtt{t}_2 \in \mathtt{s} \text{ with } \mathtt{t}[\mathtt{X}_1] = \mathtt{t}_1 \wedge \mathtt{t}[\mathtt{X}_2] = \mathtt{t}_2\}$
- **Properties**:
  - Commutative: $\mathtt{r} \bowtie \mathtt{s} = \mathtt{s} \bowtie \mathtt{r}$
  - Associative: $\mathtt{r} \bowtie (\mathtt{s} \bowtie \mathtt{t}) = (\mathtt{r} \bowtie \mathtt{s}) \bowtie \mathtt{t}$
  - Therefore, we can write n-ary joins without ambiguity:

$$\mathtt{r}_1 \bowtie \mathtt{r}_2 \bowtie \cdots \bowtie \mathtt{r}_n$$

Joins can be empty

- r:

Employee	Department
Smith	Sales
Blum	Production
Lewis	Production

- s:

Department	Head
HR	Morison

- $\mathtt{t} = \mathtt{r} \bowtie \mathtt{s}$:

Employee	Department	Head

More on Natural Join

- $\mathtt{r}(\mathtt{X}_1), \mathtt{s}(\mathtt{X}_2)$
  - No assumptions have been made w.r.t. $\mathtt{X}_1, \mathtt{X}_2$
  - They could be equal, or disjoint.
- **If they are equal:** $\mathtt{r} \bowtie \mathtt{s}$ is a relation on $\mathtt{X}_1 = \mathtt{X}_2 = \mathtt{X}$:
  - $\{\mathtt{t}$ on $\mathtt{X} \mid \mathtt{t}[\mathtt{X}] \in \mathtt{r} \wedge \mathtt{t}[\mathtt{X}] \in \mathtt{s}\} = ?$

  or, equivalently:
  - $\{\mathtt{t}$ on $\mathtt{X} \mid \exists \mathtt{t}_1 \in \mathtt{r} \wedge \exists \mathtt{t}_2 \in \mathtt{s}$ with $\mathtt{t}[\mathtt{X}] = \mathtt{t}_1 \wedge \mathtt{t}[\mathtt{X}] = \mathtt{t}_2\} = ?$
- **If they are disjoint**: $\mathtt{r} \bowtie \mathtt{s}$ is a relation on $\mathtt{X}_1 \cup \mathtt{X}_2 = \mathtt{X}$:
  - $\{\mathtt{t}$ on $\mathtt{X}_1\mathtt{X}_2 \mid \mathtt{t}[\mathtt{X}_1] \in \mathtt{r} \wedge \mathtt{t}[\mathtt{X}_2] \in \mathtt{s}\} = ?$

  or, equivalently:
  - $\{\mathtt{t}$ on $\mathtt{X}_1\mathtt{X}_2 \mid \exists \mathtt{t}_1 \in \mathtt{r} \wedge \exists \mathtt{t}_2 \in \mathtt{s}$ with $\mathtt{t}[\mathtt{X}_1] = \mathtt{t}_1 \wedge \mathtt{t}[\mathtt{X}_2] = \mathtt{t}_2\} = ?$

Theta-Join

- In most cases, a Cartesian Product is meaningul only if followed by a selection.
- **Theta-join**:

$$\mathtt{r} \bowtie_\theta \mathtt{s} = \sigma_\theta(\mathtt{r} \times \mathtt{s})$$

- **Equi-join**: when the θ condition is a conjunction of equality conditions.

Interaction of relational algebra operators

- $\pi_{\vec{\mathtt{A}}}(\mathtt{R} \cup \mathtt{S}) = \pi_{\vec{\mathtt{A}}}(\mathtt{R}) \cup \pi_{\vec{\mathtt{A}}}(\mathtt{S})$
- $\sigma_{\mathtt{c}}(\mathtt{R} \cup \mathtt{S}) = \sigma_{\mathtt{c}}(\mathtt{R}) \cup \sigma_{\mathtt{c}}(\mathtt{S})$
- $(\mathtt{R} \cup \mathtt{S}) \times \mathtt{T} = \mathtt{R} \times \mathtt{T} \cup \mathtt{S} \times \mathtt{T}$
- $\mathtt{T} \times (\mathtt{R} \cup \mathtt{S}) = \mathtt{T} \times \mathtt{R} \cup \mathtt{T} \times \mathtt{S}$

Assignment Operation

- Provides a convenient way to express complex queries.
- Write a query as a sequential program consisting of
  - a series of assignments
  - followed by an expression whose vales is displayed as a result of the query.
- Notation: $\texttt{temp} \leftarrow \texttt{query}$
- An assignment must always be made to a temporary relation variable.
  - The variable may be used in subsequent expressions.
- **Example**:
  - $\texttt{temp1} \leftarrow \texttt{r} \bowtie \texttt{s}$
  - $\texttt{temp2} \leftarrow \pi_{\texttt{A,B,C}}(\texttt{temp1})$

Division Operator

- Recall the integer division operator:
  - Given two integers X and Y, define $Z = X/Y$, so that Z*Y $= X$.
  - Can't always quite do this, so instead define Z to be the largest integer number smaller than X/Y
- Apply the same idea to the relational division operator, $\div$.
  - In that sense, $\div$ is the inverse of $\times$

Division Operator

- Suited for queries that include "for all"
- $\mathtt{r}(\mathtt{X}_1), \mathtt{s}(\mathtt{X}_2)$, where
  - $\mathtt{X}_1 = \{\mathtt{A}_1, \cdots, \mathtt{A}_m, \mathtt{B}_1, \cdots, \mathtt{B}_n\}$
  - $\mathtt{X}_2 = \{\mathtt{B}_1, \cdots, \mathtt{B}_n\}$
- Notation: $\mathrm{r} \div \mathrm{s}$
- $\mathrm{r} \div \mathrm{s}$ consists of those rows $\mathtt{t}$ such that for every row $\mathtt{u} \in \mathtt{s}$, the row resulting from concatenating $\mathtt{t}$ and $\mathtt{u}$ can be found in relation r, and there is no larger set of rows for which this is true.
  - The schema of $\mathrm{r} \div \mathrm{s}$ is $(\mathtt{A}_1, \cdots, \mathtt{A}_m)$

Division Operator

- Semantics: $\mathtt{r} \div \mathtt{s} = \{\mathtt{t} \mid \mathtt{t} \in \pi_{\mathtt{X}_1 - \mathtt{X}_2}(\mathtt{r}) \ \wedge \ \forall \mathtt{u} \in \mathtt{s}\ (\mathtt{tu} \in \mathtt{r})\}$
  - The schema of $\mathrm{r} \div \mathrm{s}$ is $(\mathtt{A}_1, \cdots, \mathtt{A}_\mathtt{m})$
  - $\mathtt{tu}$ means the concatenation of tuple t with tuple u

Division Operator Examples

- Consider R:

A	B	C
a_1	b_1	c_1
a_2	b_1	c_1
a_1	b_2	c_1
a_1	b_2	c_2
a_2	b_1	c_2
a_1	b_2	c_3
a_1	b_2	c_4
a_1	b_1	c_5

and S:

C
c_1

- We want $R \div S$

Division Operator Examples

- Consider R:

A	B	C
a_1	b_1	c_1
a_2	b_1	c_1
a_1	b_2	c_1
a_1	b_2	c_2
a_2	b_1	c_2
a_1	b_2	c_3
a_1	b_2	c_4
a_1	b_1	c_5

and S:

C
c_1

- We want $R \div S$

A	B
a_1	b_1
a_2	b_1
a_1	b_2

Division Operator Examples

- Consider R:

A	B	C
a_1	b_1	c_1
a_2	b_1	c_1
a_2	b_1	c_2
a_1	b_2	c_3
a_1	b_2	c_1
a_1	b_2	c_2
a_1	b_2	c_4
a_1	b_1	c_5

and S:

C
c_1
c_2

- We want $R \div S$

Division Operator Examples

- Consider R:

A	B	C
a_1	b_1	c_1
a_2	b_1	c_1
a_2	b_1	c_2
a_1	b_2	c_3
a_1	b_2	c_1
a_1	b_2	c_2
a_1	b_2	c_4
a_1	b_1	c_5

and S:

C
c_1
c_2

- We want $R \div S$

A	B
a_2	b_1
a_1	b_2

Division Operator Examples

- Consider R:

A	B	C
a_1	b_1	c_1
a_2	b_1	c_1
a_2	b_1	c_2
a_1	b_2	c_3
a_1	b_2	c_1
a_1	b_2	c_2
a_1	b_2	c_4
a_1	b_1	c_5

and S:

B	C
b_1	c_1

- We want $R \div S$

Division Operator Examples

- Consider R:

A	B	C
a_1	b_1	c_1
a_2	b_1	c_1
a_2	b_1	c_2
a_1	b_2	c_3
a_1	b_2	c_1
a_1	b_2	c_2
a_1	b_2	c_4
a_1	b_1	c_5

and S:

B	C
b_1	c_1

- We want $R \div S$

A
a_1
a_2

Division Operator Examples

Movies:

mID	title	director	year	length
1	Shining	Kubrick	1980	146
2	Player	Altman	1992	146
3	Chinatown	Polanski	1974	131
4	Repulsion	Polanski	1965	143
5	Star Wars IV	Lucas	1977	126
6	American Graffiti	Lucas	1973	110
7	Full Metal Jacket	Kubrick	1987	156

Artists:

aID	aName
1	Jack Nicholson
2	Harrison Ford
3	Stone
4	Fisher

Roles:

mID	aID	character
1	1	Jack Torrance
3	1	Jake 'J.J.' Gittes
1	3	Delbert Grady
5	2	Han Solo
6	2	Bob Falfa
5	4	Princess Leia Organa

- We want to find "actors who played in **all** Lucas's movies".

Division Operator Examples

Strategy:

- Let S be the set of all Lucas's movies:

$$\pi_{\mathrm{mID}}(\sigma_{\mathrm{director}='\mathrm{Lucas}'}(\mathrm{Movies}))$$

- Let R be the set of all tuples `(mID, aID)` of actors (`aID`) playing in movies:

$$\pi_{\mathrm{mID,aID}}(\mathrm{Roles})$$

- Let T be the set of actors who played in **all** Lucas's movies.
  - How do we obtain it?

Division Operator Examples

Strategy:

- Let S be the set of all Lucas's movies:

$$\pi_{\mathrm{mID}}(\sigma_{\mathrm{director}=\text{'Lucas'}}(\mathrm{Movies}))$$

- Let R be the set of all tuples `(mID, aID)` of actors (`aID`) playing in movies:

$$\pi_{\mathrm{mID,aID}}(\mathrm{Roles})$$

- Let T be the set of actors who played in **all** Lucas's movies.
  - How do we obtain it?
  - $\mathrm{R} \div \mathrm{S}$
  - $\pi_{\mathrm{mID,aID}}(\mathrm{Roles}) \div \pi_{\mathrm{mID}}(\sigma_{\mathrm{director}=\text{'Lucas'}}(\mathrm{Movies})$

Division Operator Examples

- Recall: S is the set of all Lucas's movies:

$$\pi_{\text{mID}}(\sigma_{\text{director}='\text{Lucas}'}(\text{Movies}))$$

- Are the 2 queries below semantically equivalent?
  - i.e., do they return the same result?

1. $\pi_{\text{mID,aID}}(\text{Roles}) \div S$

2. $\pi_{\text{aID}}(\text{Roles} \div S)$

More on Relational Algebra Expressions

- A basic expression in the relational algebra consists of either one of the following:
  - A relation in the database
  - A constant relation

More on Relational Algebra Expressions

- Let E_1 and E_2 be RA expressions;
- the following are all RA expressions:
  - $\mathrm{E}_1 \cap \mathrm{E}_2$
  - $\mathrm{E}_1 \cup \mathrm{E}_2$
  - $\mathrm{E}_1 - \mathrm{E}_2$
  - $\mathrm{E}_1 \times \mathrm{E}_2$
  - $\mathrm{E}_1 \bowtie \mathrm{E}_2$
  - $\mathrm{E}_1 \div \mathrm{E}_2$
  - $\sigma_{\mathrm{cond}}(\mathrm{E}_1)$, where cond is a condition on attributes of E_1
  - $\pi_{\mathrm{attr}}(\mathrm{E}_1)$, where attr is a subset of attributes in E_1
  - $\rho_{\mathrm{new} \leftarrow \mathrm{old}}(\mathrm{E}_1)$