The Relational Data Model: Outline

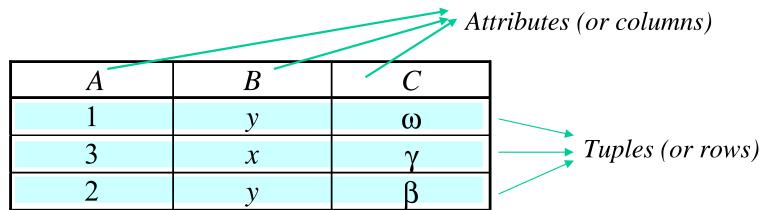
- Structure and Definitions
 - Relation, schema, tuple
 - Domain, attributes
- Relational Algebra
- Summary

Structure and Definitions

- Given sets A_1, A_2, \dots, A_n , a relation r is a subset of $A_1 \times A_2 \times \dots \times A_n$
 - \bullet *n* is also often called *dimensionality* of r.
- $(A_1, A_2, ..., A_n)$ is termed the relation's schema.
- A relation is a set of *n*-tuples $(a_1, a_2, ..., a_n)$ where $a_i \in A_i$
- The *cardinality* of a relation is the number of tuples in it.

Example

- Let
 - $A = \{1, 2, 3\}$
 - $B = \{x, y, z\}$
 - $C = \{\alpha, \beta, \gamma, \omega\}$
- Then $r = \{(1, y, \omega), (3, x, \gamma), (2, y, \beta)\}$ is a relation over $A \times B \times C$, its cardinality is 3.
- The schema of r is R = (A, B, C)
- Generally we will specify relations by tables.



More Notations

- Sets $A_1, A_2, ..., A_n$ are domains; their names are attributes.
- $R = (A_1, A_2, ..., A_n)$ is a relation schema.
- r(R) is a *relation* on the relation schema R.
- An element *t* of *r* is a *tuple*.
- We refer to component values of a tuple t by $t[A_i] = v_i$ (the value of attribute A_i for tuple t).
- $t[A_i, ..., A_k]$ refers to the *subtuple* of t containing the values of attributes $A_i, ..., A_k$ respectively.

Keys

- Let $K \subseteq R$
- K is a *superkey* of R if values for K are sufficient to identify a unique tuple of each possible relation r(R)
- Example: Person = (name, street, city)
 - {name, street} and {name} are both superkeys of Person, if no two persons can have the same name
- *K* is a *candidate key* if *K* is minimal
 - Example: {name} is a candidate key for *Person*, since it is a superkey and no subset of it is a superkey.

Primary key And Foreign Keys

- *Primary key* is a candidate key chosen as the principal means of identifying tuples within a relation
 - Should choose an attribute whose value never, or very rarely, changes.
 - E.g., Person = (name, street, city, email)
 - *name* can be a primary key, if no two persons can have the same name
 - Although email address is unique, it may change often
- A schema may have an attribute that corresponds to the primary key of another relation. The attribute is called a *foreign key*.
 - Only values occurring in the primary key attribute of the *referenced* relation may occur in the foreign key attribute of the *referencing relation*.
 - E.g., Grade = (name, course, grade)
 - Here, *name* attribute is a foreign key to *Person*

Characteristics of Relations

- Ordering of tuples in a relation r(R)
 - The tuples are not considered to be ordered, even though they appear to be in a tabular form.
- Ordering of attributes in a relation schema *R* (and of values within each tuple)
 - We will consider the attributes in $R(A_1, A_2, ..., A_n)$ and the values $t = \langle v_1, v_2, ..., v_n \rangle$ to be ordered.
- Values in a tuple
 - All values are considered atomic (indivisible). A special *null* value is used to represent values that are unknown or inapplicable to certain tuples.

The Relational Data Model: Outline

- Structure and Definitions
- Relational Algebra
 - Operations
 - A Real-Life Example
 - Limitations
- Summary

Relational Algebra

- Six primary operators
 - Selection: σ
 - Projection: π
 - Union: \cup
 - Difference: –
 - Cartesian Product: ×
 - Rename: ρ
- Convenient derived operators
 - Intersection: ∩
 - Theta join \bowtie_{θ} , equijoin $\bowtie_{=}$, and natural join \bowtie
 - Semijoins: left \bowtie and right \bowtie
 - Relational division: ÷
- Each operator takes one or two relations as input, and produces a new relation as the result

Selection

$$\sigma_P(r) = \{t \mid t \in r \land P(t) \}$$

- P is a formula in propositional calculus, dealing with terms of the form:
 - attribute (or constant) = attribute (or constant)
 - attribute ≠ attribute
 - attribute < attribute
 - attribute ≤ attribute
 - attribute ≥ attribute
 - attribute > attribute
 - term ∧ term (conjunction)
 - term v term (disjunktion)
 - ¬ term (negation)

Selection Example

r:

A	B	C
1	у	ω
3	X	γ
2	у	β

$$\sigma_{B=y \wedge A>1}(r)$$
:

A	B	C
2	y	β

Projection

$$\pi_X(r) = \{t[X] \mid t \in r\}$$

- Let $X = \{A_1, A_2, ..., A_n\}$
- The result is a relation of *n* columns obtained by removing the columns that are not specified.
- Example

r:	A	В	C
- •	1	\mathcal{Y}	ω
	3	\boldsymbol{x}	γ
	2	У	β

$\pi_{A,C}(r)$:	A	C
$A,C \leftarrow \gamma$	1	ω
	3	γ
	2	β

$\pi_B(r)$:	В
$D \vee \gamma$	У
	X

Union

$$r \cup s = \{t \mid t \in r \lor t \in s\}$$

- Assume
 - r and s have the same arity (the number of attributes).
 - The attributes of *r* and *s* are compatible.
- Example

740	A	B	C
r:	1	y	ω
	3	\mathcal{X}	γ
	2	y	β

c.	A	B	C
5 .	4	W	ζ
	3	\mathcal{X}	γ

	A	В	\boldsymbol{C}
$r \cup s$:	1	y	ω
	3	X	γ
	2	y	β
	4	W	ζ

Difference

$$r - s = \{t \mid t \in r \land \neg (t \in s)\}$$

- Assume
 - *r* and *s* have the same arity.
 - The attributes of r and s are compatible.

Example

740	\boldsymbol{A}	B	\boldsymbol{C}
r:	1	y	ω
	3	\mathcal{X}	γ
	2	ν	β

~•	A	B	C
<i>S</i> :	4	W	ζ
	3	\mathcal{X}	γ

r-s:

A	В	\boldsymbol{C}
1	у	ω
2	y	β

Cartesian Product

$$r \times s = \{t \circ q \mid t \in r \land q \in s\}$$

- o is concatenation
- The schema of the resulting relation is $R \circ S$

 $P: \begin{array}{c|cccc} A & B & C \\ \hline 1 & y & \omega \\ \hline 3 & x & \gamma \\ \hline 2 & y & \beta \end{array}$

 D
 A

 "Tom"
 1

 "Eric"
 2

 $r \times s$:
 A
 B
 C
 D
 A

 1
 y
 ω "Tom"
 1

 3
 x
 γ "Tom"
 1

 2
 y
 β "Tom"
 1

 1
 y
 ω "Eric"
 2

 3
 x
 γ "Eric"
 2

 2
 y
 β "Eric"
 2

Rename

- Allows us to name, and therefore to refer to, the results of relational-algebra expressions.
- Allows us to refer to a relation by more than one name.
- Example:
 - $\rho_{X}(E)$ returns the expression E under the name X
- If a relational-algebra expression E has arity n, then

$$\rho_{x(A_1,A_2,\ldots,A_n)}(E)$$

returns the result of expression E under the name X, and with the attributes renamed to $A_1, A_2, ..., A_n$.

Intersection

$$r \cap s = \{t \mid t \in r \land t \in s\} = r - (r - s)$$

- Assume
 - *r* and *s* have the same arity.
 - Attributes of *r* and *s* are compatible

Example

140	A	B	C
r:	1	y	ω
	3	\mathcal{X}	γ
	2	y	β

G •	A	B	C
<i>S</i> :	4	W	ζ
	3	\overline{x}	γ

r		S	•
	1 1	1)	_

A	В	C
3	\mathcal{X}	γ

Joins

- Joins are Cartesian products coupled with selections and projections.
- Theta join
 - $r \bowtie_{\theta} s = \sigma_{\theta} (r \times s)$
 - Example: $r \bowtie_{A < E} s = \sigma_{A < E} (r \times s)$
- Equijoin
 - A theta join in which θ is an equality predicate
 - Example: $r \bowtie_{A=E} s = \sigma_{A=E} (r \times s)$
- Natural join ⋈
- Semijoins
 - Left semijoin ×
 - Right semijoin ×

Theta Join

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

Example

r·	A	B	\boldsymbol{C}
' •	1	y	ω
	0	\mathcal{X}	γ
	2	y	β

$$r \bowtie_{A < E} S$$
:
 $A = B = C = D = E$

 1
 y
 w
 "Tom"
 3

 0
 x
 γ
 "Tom"
 3

 0
 x
 γ
 "Eric"
 1

 2
 y
 β
 "Tom"
 3

Equijoin

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

- An equijoin is a special theta join in which θ is an equality predicate
- Example: $r \bowtie_{A=E} s$ (A and E must be compatible)

r·	A	B	C
•	1	y	ω
	0	\mathcal{X}	γ
	2	y	β

s:	D	E
.	"Tom"	3
	"Eric"	1

$r\bowtie_{A-F} S$:	A	В	C	D	E
A = E	1	у	ω	"Eric"	1

• Usually only A or E is retained as they are the same

Natural Join

$$r\bowtie s=\pi_{R\cup S}(r\bowtie_= s)$$

- Schema of result is $R \cup S$
- Let t be a tuple in the result. On each of the attributes in $R \cap S$,
 - t[R] has the same value as a tuple $t_R \in r$.
 - t[S] has the same value as a tuple $t_s \in s$.
- Example

/^ •	A	В	C
•	1	у	ω
	3	$\boldsymbol{\mathcal{X}}$	γ
	2	y	β

s:	D	В
D •	"Tom"	y
	"Eric"	$\boldsymbol{\mathcal{X}}$

1	\bowtie	C	•
		S	•

A	В	C	D
1	y	ω	"Tom"
3	X	γ	"Eric"
2	y	β	"Tom"

Another Natural Join Example

• Relations r, s:

Α	В	С	D	
α	1	α	а	
$\mid \beta \mid$	2	γ	а	
$ \gamma $	4	β	b	
$\mid \alpha \mid$	1	γ	а	
δ	2	β	b	
r				

В	D	Ε
1 3 1 2 3	a a a b b	$\begin{array}{c} \alpha \\ \beta \\ \gamma \\ \delta \\ \in \end{array}$

■ r⋈s

A	В	С	D	E
α	1	α	а	α
α	1	α	а	γ
α	1	γ	а	α
α	1	γ	а	γ
δ	2	β	b	δ

Semijoins

$$r \bowtie s = \pi_R (r \bowtie s)$$

- The result has the same schema as the left-hand argument, *r*.
- Example

740	A	В	\boldsymbol{C}
r:	1	У	ω
	3	\mathcal{X}	γ
	2	у	β

c •	D	B
D •	"Tom"	У

1	X	C	•
/		S	•

A	В	C
1	y	ω
2	y	β

• Right semijoin: $r \bowtie s = \pi_S (r \bowtie s)$

Relational Division

- Let *r* and *s* be relations on schemes *R* and *S* respectively, where
 - $R = (A_1, A_2, ..., A_n, B_1, B_2, ..., B_n)$
 - $S = (B_1, B_2, ..., B_n)$
- The result of r divided by s is a relation on scheme $R \div S = (A_1, A_2, ..., A_n)$
- $r \div s = \{ t \mid t \in \pi_{R-S}(r) \land \forall u \in s (t \circ u \in r) \}$
- Property
 - $q = r \div s$ is the largest relation satisfying $q \times s \subseteq r$
 - Given a tuple *t* in the result, for any tuple *u* from *s*, their concatenation is a tuple in *r*

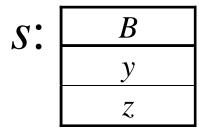
Relational Division Example 1

$$r \div s = \{ t \mid t \in \pi_{R-S}(r) \land \forall u \in s (t \circ u \in r) \}$$

• Given a tuple *t* in the result, for any tuple *u* from *s*, their concatenation is a tuple in *r*

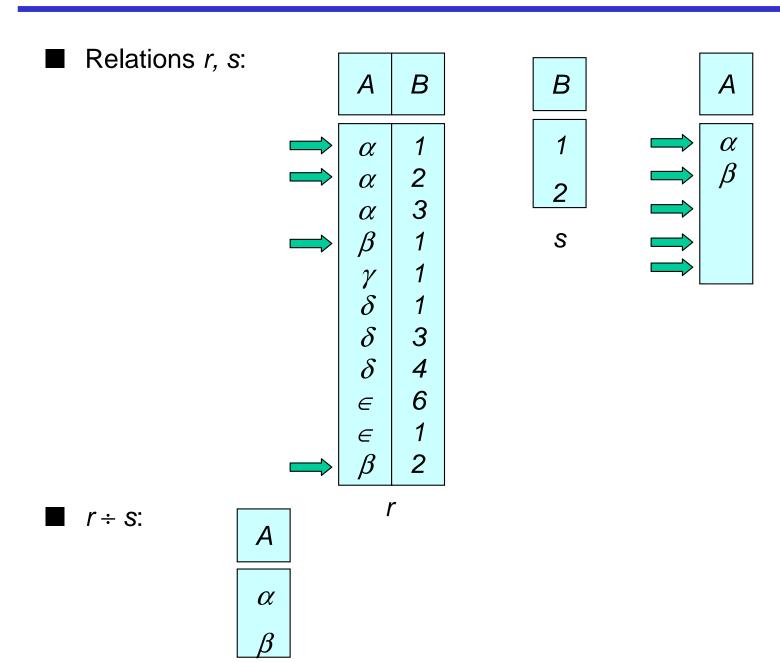
Example

7.0	A	В	C
r:	1	y	ω
	3	\mathcal{X}	γ
	2	y	β
	1	z	ω



$r \div s$:	A	C
	1	ω

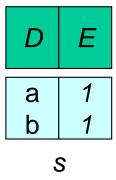
Relational Division Example 2



Relational Division Example 3

Relations *r*, *s*:

Α	В	С	D	E
α	а	α	а	1
α	а	γ	а	1
α	а	γ	b	1
β	а	γ	а	1
β		γ	b	3
γ	a a	γ	а	1
$egin{array}{c} lpha \\ lpha \\ eta \\ eta \\ \gamma \\ \gamma \\ \gamma \end{array}$	а	γ γ γ γ γ	b	1
γ	а	β	b	1
r				



 $r \div s$:

Α	В	С
$\alpha \\ \gamma$	a a	$\gamma \\ \gamma$

Video Store Schema

- Customer (<u>CustomerID</u>, Name, Street, City, State)
- Film (FilmID, Title, RentalPrice, Kind)
- Reserves (CustomerID, FilmID, ResDate)

- Underlined attributes together form primary keys
- In Reserves relation, both <u>CustomerID</u> and <u>FilmID</u> are foreign keys

Video Store Queries

• List the information for films with a rental price over \$4.

$$\sigma_{\text{RentalPrice} > 4}(\text{Film})$$

• List the titles of films with a rental price over \$4.

$$\pi_{\text{Title}} \sigma_{\text{RentalPrice} > 4} (\text{Film})$$

• List the outrageously priced films (over \$4 or under \$1).

$$\pi_{\text{Title}} \left(\sigma_{\text{RentalPrice} > 4}(\text{Film}) \cup \sigma_{\text{RentalPrice} < 1}(\text{Film}) \right)$$

• List the IDs of the expensive films that have not been reserved.

$$\pi_{\text{FilmId}} (\sigma_{\text{RentalPrice} > 4}(\text{Film})) - \pi_{\text{FilmId}}(\text{Reserves})$$

• List the titles of all reserved films.

$$\pi_{\text{Title}}$$
 ($\sigma_{\text{Film.FilmID} = \text{Reserves.FilmID}}$ (Film × Reserves))

 π_{Title} (Film × Reserves)

• List the customers who have reserved a film.

$$\pi_{\text{Name}}$$
 ($\sigma_{\text{Customer.CustomerID}} = \text{Reserves.CustomerID}$ (Customer × Reserves))

$$\pi_{\text{Name}}$$
 (Customer \bowtie Reserves)

List the customers who have reserved expensive films.

```
\pi_{\text{Name}} (\sigma_{\text{RentalPrice} > 4} (Customer \bowtie Reserves \bowtie Film)) \pi_{\text{Name}} (Customer \bowtie \sigma_{\text{RentalPrice} > 4} (Reserves \bowtie Film)) \pi_{\text{Name}} (Customer \bowtie Reserves \bowtie \sigma_{\text{RentalPrice} > 4} (Film))
```

• List the streets of customers who have reserved foreign films.

$$\pi_{\text{Street}}$$
 (Customer \bowtie Reserves $\bowtie \sigma_{\text{Kind} = \text{"F"}}$ (Film))

- List the customers who have reserved *all* the foreign films.
 - Identify the foreign films.

$$\pi_{\text{FilmID}} \left(\sigma_{\text{Kind} = \text{"F"}} \left(\text{Film} \right) \right)$$

Identify those customers who have reserved all foreign films.

$$\pi_{\text{CustomerID,FilmID}} (\text{Reserves}) \div \\ \pi_{\text{FilmID}} (\sigma_{\text{Kind} = \text{"F"}} (\text{Film}))$$

Now figure out the names of those customers.

$$\pi_{\text{Name}}$$
 (Customer \bowtie ($\pi_{\text{CustomerID,FilmID}}$ (Reserves) \div π_{FilmID} ($\sigma_{\text{Kind} = \text{``F''}}$ (Film))))

- Find the film(s) with the highest rental price.
- We need a renaming operator: ρ_{Name}
- Alternative formulation: Find the film(s) with a rental price for which no other rental price is higher.

$$\pi_{\text{Title}}(\text{Film}) - \\ \pi_{\text{F2.Title}}(\sigma_{\text{Film.RentalPrice}} > \text{F2.RentalPrice} (\text{Film} \times \rho_{\text{F2}}(\text{Film})))$$

Relational Completeness

- All the operators can be expressed in terms of the six basic operators: σ , π , -, \times , \cup , ρ
- This set is called a *complete set* of relational algebraic operators.
- Any query language that is at least as powerful as these operators is termed (query) relationally complete.

Limitations of the Algebra

- Can't do aggregates.
 - How many films has each customer reserved?
- Can't handle "missing" data.
 - Make a list of the films, along with who reserved it, if applicable.
- Can't perform transitive closure.
 - For a part of(Part, ConstituentPart) relation, find all parts in the car door.
- Can't sort, or print in various formats.
 - Print a reserved summary, sorted by customer name.
- Can't modify the database.
 - Increase all \$3.25 rentals to \$3.50.

Outer Joins

- In a regular equijoin or natural join, tuples in *r* or *s* that do not having matching tuples in the other relation do not appear in the result.
- The outer joins retain these tuples, and place nulls in the missing attributes.
- Left outer join:

$$r \times s = r \bowtie s \cup ((r - (r \bowtie s)) \times (null, ..., null))$$

• Right outer join:

$$r \times s = r \bowtie s \cup ((null, ..., null) \times (s - (s \bowtie r)))$$

• Full outer join:

$$r \times s = r \bowtie s \cup ((r - (r \bowtie s)) \times (null, ..., null))$$

 $\cup ((null, ..., null) \times (s - (s \bowtie r)))$

Outer Join Examples

v·	A	B	C
/.	1	у	ω
	3	X	γ
	2	\mathcal{Z}	В

D	В
"Tom"	у
"Eric"	X
"Melanie"	W

			 -	
$r \times s$:	A	В	C	D
	1	y	ω	"Tom"
	3	\mathcal{X}	γ	"Eric"
	2	\overline{z}	β	NULL

$r \times s$:	A	В	C	D
	1	y	ω	"Tom"
	3	\mathcal{X}	γ	"Eric"
	NULL	W	NULL	"Melanie"

$r \times s$:	A	B	\boldsymbol{C}	D
	1	у	ω	"Tom"
	3	\mathcal{X}	γ	"Eric"
	2	$\boldsymbol{\mathcal{Z}}$	β	NULL
	NULL	\mathcal{W}	NULL	"Melanie"

The Relational Data Model: Outline

- Structure and Definitions
- Relational Algebra Operations
- Summary

Summary

- Data model
 - Duplicates are not allowed
- Relational algebra
 - Objects are relations (sets of n-tuples).
- Operators
 - Only six basic operators
 - Many derived operators
 - Each operator requires input relation(s) and produces a resulting relation
- Operators are the basis for query processing and optimization