The Relational Data Model: Outline

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

Structure and Definitions

- Given sets $A_1, A_2, \ldots, A_n$, a *relation* r is a subset of $A_1 \times A_2 \times \ldots \times A_n$
  - n is also often called *dimensionality* of r.
- $(A_1, A_2, \ldots, A_n)$ is termed the relation's *schema*.
- A relation is a set of *n-tuples* $(a_1, a_2, \ldots, a_n)$ 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 $R = (A, B, \text{C})$
- Generally we will specify relations by tables.

Attributes (or columns)

A	B	C
1	y	ω
3	x	γ
2	y	β

Tuples (or rows)

More Notations

- Sets $A_1, A_2, \ldots, A_n$ are *domains*; their names are *attributes*.
- $R = (A_1, A_2, \ldots, A_n)$ 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).
- $t[A_i, \ldots, A_k]$ refers to the *subtuple* of t containing the values of attributes $A_i, \ldots, A_k$ 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 $(A_1, A_2, \ldots, A_n)$ and the values $t = \langle v_1, v_2, \ldots, v_n \rangle$ 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: $\times$
  - Rename: ρ
- Convenient derived operators
  - Intersection: $\cap$
  - Theta join $\bowtie_\theta$, equijoin $\bowtie_=$, and natural join $\bowtie$
  - Semijoins: left $\ltimes$ and right $\rtimes$
  - Relational division: $\div$
- Each operator takes one or two relations as input, and produces a new relation as the result

Selection

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

- P is a formula in propositional calculus, dealing with terms of the form:
  - attribute (or constant) = attribute (or constant)
  - attribute $\neq$ attribute
  - attribute $<$ attribute
  - attribute $\leq$ attribute
  - attribute $\geq$ attribute
  - attribute $>$ attribute
  - term $\wedge$ term (conjunction)
  - term $\vee$ term (disjunktion)
  - $\neg$ term (negation)

Selection Example

r:

A	B	C
1	y	ω
3	x	γ
2	y	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 $X = \{A_1, A_2, \ldots, A_n\}$
- The result is a relation of n columns obtained by removing the columns that are not specified.
- Example

r:

A	B	C
1	y	ω
3	x	γ
2	y	β

$\pi_{A,C}(r)$:

A	C
1	ω
3	γ
2	β

$\pi_B(r)$:

B
y
x

Union

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

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

r:

A	B	C
1	y	ω
3	x	γ
2	y	β

s:

A	B	C
4	w	ζ
3	x	γ

$r \cup s$:

A	B	C
1	y	ω
3	x	γ
2	y	β
4	w	ζ

Difference

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

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

r:

A	B	C
1	y	ω
3	x	γ
2	y	β

s:

A	B	C
4	w	ζ
3	x	γ

$r - s$:

A	B	C
1	y	ω
2	y	β

Cartesian Product

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

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

r:

A	B	C
1	y	ω
3	x	γ
2	y	β

s:

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 $A_1, A_2, \ldots., A_n$.

Intersection

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

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

r:

A	B	C
1	y	ω
3	x	γ
2	y	β

s:

A	B	C
4	w	ζ
3	x	γ

$r \cap s$:

A	B	C
3	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 $\bowtie$
- Semijoins
  - Left semijoin $\ltimes$
  - Right semijoin $\rtimes$

Theta Join

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

- Example

r:

A	*B*	*C*
1	*y*	ω
0	*x*	γ
2	*y*	β

s:

D	*E*
“Tom”	3
“Eric”	1

$r \bowtie_{A < E} s$:

A	*B*	*C*	*D*	*E*
1	*y*	ω	“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	x	γ
2	y	β

s:

D	E
“Tom”	3
“Eric”	1

$r \bowtie_{A=E} s$:

A	B	C	D	E
1	y	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

r:

A	B	C
1	y	ω
3	x	γ
2	y	β

s:

D	B
"Tom"	y
"Eric"	x

$r \bowtie s$:

A	B	C	D
1	y	ω	"Tom"
3	x	γ	"Eric"
2	y	β	"Tom"

Another 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	$\in$

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

Semijoins

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

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

r:

A	B	C
1	y	ω
3	x	γ
2	y	β

s:

D	B
“Tom”	y

$r \ltimes s$:

A	B	C
1	y	ω
2	y	β

- Right semijoin: $r \rtimes s = \pi_S\,(r \bowtie s)$

Relational Division

- Let *r* and *s* be relations on schemes *R* and *S* respectively, where
  - $R = (A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_n)$
  - $S = (B_1, B_2, \ldots, B_n)$
- The result of *r* divided by *s* is a relation on scheme $R \div S = (A_1, A_2, \ldots, A_n)$
- $r \div s = \{\, t \mid t \in \pi_{R-S}(r) \wedge \forall\, u \in s\, (\, t \circ u \in r\,)\, \}$
- 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

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

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

- Example

r:

A	*B*	*C*
1	*y*	ω
3	*x*	γ
2	*y*	β
1	*z*	ω

s:

B
y
z

$r \div s$:

A	*C*
1	ω

Relational Division Example 2

- Relations *r, s*:

A	B
α	1
α	2
α	3
β	1
γ	1
δ	1
δ	3
δ	4
$\in$	6
$\in$	1
β	2

r

B
1
2

s

A
α
β

- $r \div s$:

A
α
β

Relational Division Example 3

- 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* ÷ *s*:

A	*B*	*C*
α	a	γ
γ	a	γ

Video Store Schema

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

- 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.

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

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

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

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

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

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

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

Video Store Queries, cont.

- List the titles of all reserved films.

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

$$\pi_{Title}\ (Film \bowtie Reserves)$$

- List the customers who have reserved a film.

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

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

Video Store Queries, cont.

- List the customers who have reserved expensive films.

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

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

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

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

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

Video Store Queries, cont.

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

$$\pi_{\text{FilmID}} (\sigma_{\text{Kind} = \text{“F”}} (\text{Film}))$$

  - 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.

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

Video Store Queries, cont.

- 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.

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

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:

$$r ⟕ s = r \bowtie s \cup ((r - (r \ltimes s)) \times (null, \ldots, null))$$

- Right outer join:

$$r ⟖ s = r \bowtie s \cup ((null, \ldots, null) \times (s - (s \ltimes r)))$$

- Full outer join:

$$r ⟗ s = r \bowtie s \cup ((r - (r \ltimes s)) \times (null, \ldots, null)) \\ \cup ((null, \ldots, null) \times (s - (s \ltimes r)))$$

Outer Join Examples

r:

A	B	C
1	*y*	ω
3	*x*	γ
2	*z*	β

s:

D	B
"Tom"	*y*
"Eric"	*x*
"Melanie"	*w*

$r ⟕ s$:

A	B	C	D
1	*y*	ω	"Tom"
3	*x*	γ	"Eric"
2	*z*	β	NULL

$r ⟖ s$:

A	B	C	D
1	*y*	ω	"Tom"
3	*x*	γ	"Eric"
NULL	*w*	NULL	"Melanie"

$r ⟗ s$:

A	B	C	D
1	*y*	ω	"Tom"
3	*x*	γ	"Eric"
2	*z*	β	NULL
NULL	*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