DATABASE SYSTEMS

Dr. Raj Sunderraman

Department of Mathematics and Computer Science
Georgia State University

January 29, 1999

Ch 6. Relational Data Model

```
DOMAIN: D is a set of atomic values.
  e.g. USA_Phone_Numbers: set of 10-digit phone numbers valid in USA
       Valid_Age: 0 to 150
       Grade_Point_Averages: real numbers between 0 and 4

RELATION SCHEMA: R, denoted by R(A1,...,An), consists of
   a relation name R and list of ATTRIBUTES A1, ..., An
   n = DEGREE of relation

Each attribute is associated with domain, dom(Ai)
```

```
  e.g. STUDENT(Name, SSN, HomePhone, Address, OfficePhone, Age, GPA)

   is a relation schema of degree = 7

A RELATION (or RELATION INSTANCE) r on relation schema R(A1,...,An)
also denoted as r(R) is a finite subset of
   dom(A1) X ... X dom(An)

Each element of a relation, such as (a1, ..., an) is called a TUPLE.
```

A RELATIONAL DATABASE SCHEMA consists of

- a set of relation schemas R1,...,Rm
- a set of Integrity constraints, IC (to be discussed later)

A RELATIONAL DATABASE INSTANCE is a set of relation instances DB = {r1,...,rn} such that each ri in an instance of Ri satisfying the constraints specified in IC.

Characteristics of Relations:

- Ordering of tuples in a relation: No order among the tuples since a relation is a set of tuples. (Figures 6.1, 6.2)

- Ordering of values within a tuple: two perspectives (ordered, unordered). We shall go with the ordered perspective. Skip unordered definition on Page 141.

- Atomic values in tuples: First normal form. special value in each domain called NULL.

- A relation can be interpreted as a collection of assertions or facts about the mini-world. Some facts correspond to ENTITIES and others correspond to RELATIONSHIPS.

```
NOTATION:

relation schema: R(A1,...,An)
n-tuple t in r(R) is <v1,...,vn>, where vi is in dom(Ai)
  - t[Ai] refers to vi
  - t[Au, Aw, ..., Az] refers to <vu,vw,...,vz>
Upper-case letters Q, R, S, refer to relation names in schemas
Lower-case letters q, r, s refer to relation instances
Lower-case letters t, u, v refer to tuples
We use R in R(A1,...,An) to refer to the relation instance
```

6.2 Constraints

```
(1) Domain Constraints:
     - atomic domains
     - data types such as integer, reals, strings, etc.

(2) Key Constraints:

  - Super Key: A subset, K, of attributes of R is a super key for R if
     for any two distinct tuples t1 and t2 in a (legal) relation
     instance of r of R, t1[K] <> t2[K].

  - Candidate key: (or simply a key) is a super key such that  none of
    its proper subsets is a super key.

  - Primary Key: One of the candidate keys chosen by the database
        designer.

DATABASE SCHEMA (with primary keys) and INSTANCE in Figures 6.5, 6.6
```

(3) Entity Integrity Constraint:

No primary key attributes can have a NULL value.

(4) Referential Integrity Constraint (between relations)

A set of attributes FK, of a relation schema R1 is a FOREIGN KEY of R1 if

(a) The attributes in FK have the same domain as the primary key attributes PK of another relation schema R2
FK references or refers to the relation R2.

and (b) A value of FK in a tuple t1 of R1 either occurs as a value of PK for some tuple t2 of R2 or is NULL.
The tuple t1 references or refers to tuple t2.

Figure 6.7

6.3 Update Operations

Insert, Delete, Modify

Database state is assumed to be consistent before updates,
i.e. all constraints are statisfied.

Update operations, when completed, must result in a consistent
database state (otherwise they should be rejected).

Constraints we will consider are:

- Domain constraints
- Key constraints,
- Entity integrity constraint
- Referential integrity constraints

```
Insert: (all 4 types of constraints may be violated).

Assume database of Figure 6.6

Examples of inserts:

(1) Insert <'Cecilia','F','Kolonsky','677678989','05-APR-50',
            '6357 windy Lane, Katy, TX',F,28000,null,4>
    into EMPLOYEE
    satisfies all constraints; ACCEPT IT

(2) Insert <'Alicia','J','Zelaya','999887777','05-APR-50',
            '6357 windy Lane, Katy, TX',F,28000,'987654321',4>
    into EMPLOYEE
    violates key constraint.
```

```
(3) Insert <'Cecilia','F','Kolonsky',null,'05-APR-50',
            '6357 windy Lane, Katy, TX',F,28000,null,4>
    into EMPLOYEE
    violates entity integrity constraint;

(4) Insert <'Cecilia','F','Kolonsky',677678989,'05-APR-50',
            '6357 Windswept, Katy, TX',F,28000,'987654321',7>
    into EMPLOYEE
    violates referential integrity (no dept = 7!!)
```

```
Two options if inserts are unacceptable:

(1) Reject the insert
(2) Try to correct the reason for rejection
     e.g. prompt user for value of key when null was specified
     or ask user to add dept = 7 then add the employee
```

```
Delete: can violate only referential integrity.

(1) Delete WORKS_ON where ESSN = '999887777' and PNO = 10
    ACCEPTABLE

(2) Delete EMPLOYEE where SSN = '999887777'
    not ACCEPTABLE because two rows in WORKS_ON refer to this tuple.

(3) Delete EMPLOYEE where SSN = '333445555'
    not acceptable because it is referenced by tuples in
     EMPLOYEE, DEPARTMENT, WORKS_ON, DEPENDENT!!
```

3 options to deal with unacceptable deletes:

(1) Reject it

(2) Attempt to cascade deletes by deleting tuples that reference the tuple that is being deleted. e.g. in example (2), delete the two referencing tuples from WORKS_ON

(3) Modify the referencing tuples (set to null or some other value); then delete the tuple.

Combinations are also possible.

```
Modify: Can violate all constraints.

(1) Modify Salary for employee with ssn = '999887777' to 28000
    Acceptable

(2) Modify DNO of Employee with ssn = '999887777' to 1
    Acceptable

(3) Modify DNO of Employee with ssn = '999887777' to 7
    not Acceptable; violates referential integrity (not dept=7)

(4) Modify ssn of employee with ssn='999887777' to '987654321'
    not acceptable; violates primary key and referential integrity

Modifying non-primary/non-foreign key usually do not create problems
Modifying = Delete followed by Insert; Issues discussed earlier
  apply to modify.
```

Relational Algebra

Set-theoretic operations:

Two relations are **union-compatible** if they have the same number of attributes and the domains of the corresponding attributes in the two relations are the same.

Consider two relations $r(R)$ and $s(S)$ that are union-compatible (normally R = S).

Union: $r \cup s = \{t | t \in r \text{ or } t \in s\}$.

Difference: $r - s = \{t | t \in r \text{ and } t \notin s\}$

Intersection: $r \cap s = \{t | t \in r \text{ and } t \in s\}$

Cartesian Product: $r(R)$ and $s(S)$ on any schemes R and S.

$$r \times s = \{t_1.t_2 | t_1 \in r \text{ and } t_2 \in s\},$$

where, $t1.t2$ is the concatenation of tuples t_1 and t_2 to form a larger tuple.

Example: set operations

r

A	B
a	b
a	c
b	d

s

A	B
a	c
a	e

$r \cup s$

A	B
a	b
a	c
b	d
a	e

$r - s$

A	B
a	b
b	d

$r \cap s$

A	B
a	c

$r \times s$

r.A	r.B	s.A	s.B
a	b	a	c
a	b	a	e
a	c	a	c
a	c	a	e
b	d	a	c
b	d	a	e

Relation-theoretic operations

Consider $r(R)$ and $s(S)$, two relations, where $R = (A_1, ..., A_n)$ and $S = (B_1, ..., B_m)$

Rename: $r(C_1, ..., C_n) = \{t | t \in r\}$ with schema $(C_1, ..., C_n)$.

Select: $\sigma_F(r) = \{t | t \in r$ and t satisfies $F\}$.

where F is a selection criteria involving constants and attributes of r. (will discuss in examples how F is constructed)

Project: $\pi_{D_1,...,D_p}(r) = \{t[D_1, ..., D_p] | t \in r\}$

where D_i is one of $A_1, ..., A_n$.

theta-Join: $r \bowtie_F s = \{t | (\exists u \in r)(\exists v \in s)(t = u.v$ and F is satisfied by u and $v)\}$

where F is a conjunction of formulas relating attributes of r with attributes of s. (will discuss in examples how F is constructed)

Natural Join: $r \bowtie s = \{t | (\exists u \in r)(\exists v \in s)(t[R] = u$ and $t[S] = v)\}$

Division: Assume $B_1, ..., B_m \subset A_1, ..., A_n$.

$$r \div s = \{t | (\forall u \in s)(t.u \in r)\}$$

Examples: relation-theoretic operations

r

A	C	D
a	c	d
a	e	f
a	g	h
b	c	d
b	g	h
c	c	d
c	e	f

$\sigma_{A='b' \text{ or } C='c'}(r)$

A	C	D
a	c	d
b	c	d
b	g	h
c	c	d

$\pi_A(r)$

A
a
b
c

r

A	C	D
a	c	d
a	e	f
a	g	h
b	c	d
b	g	h
c	c	d
c	e	f

t

B	C	D
b	c	d
b	e	f

$r \bowtie_{r.A=t.B} t$

A	C	D	B	C	D
b	c	d	b	c	d
b	c	d	b	e	f
b	g	h	b	c	d
b	g	h	b	e	f

r

A	C	D
a	c	d
a	e	f
a	g	h
b	c	d
b	g	h
c	c	d
c	e	f

t

B	C	D
b	c	d
b	e	f

$r \bowtie t$

A	C	D	B
a	c	d	b
a	e	f	b
b	c	d	b
c	c	d	b
c	e	f	b

r

A	C	D
a	c	d
a	e	f
a	g	h
b	c	d
b	g	h
c	c	d
c	e	f

s

C	D
c	d
e	f

$r \div s$

A
a
c

Basic Relational Algebra Operations

- Basic set: union, difference, Cartesian product, rename, select, and project.
- none of them can be expressed in terms of the others.
- intersection, theta-join, natural join, and division can be expressed in terms of the basic operators as follows:

Intersection: $r \cap s = r - (r - s)$

theta Join: $r \bowtie_F s = \sigma_F(r \times s)$

Natural Join: $r \bowtie s = \pi_{R \cap S}(\sigma_F(r \times s))$

where F is a selection condition which indicates that the tuple values under the common attributes of r and s are equal.

Division: $r \div s = \pi_{R-S}(r) - \pi_{R-S}((\pi_{R-S}(r) \times s) - r)$

Even though relation schemes are defined as sequences, they are treated as sets in these equalities for simplicity.

An explanation for the equality for division is in order!

- First, all candidate tuples for the result are calculated by the expression

  $$\pi_{R-S}(r)$$

- Next, these candidate tuples are combined with all tuples of s in the following expression

  $$\pi_{R-S}(r) \times s$$

  to give a relation containing all combinations of candidate tuples with all tuples of s.

- Since we are looking for tuples under the scheme $R - S$ which combine with all tuples of s and are also present in r, if we subtract r from the previous expression, we will get all the combinations of tuples that are "missing" in r.

  $$(\pi_{R-S}(r) \times s) - r$$

- By projecting these tuples on $R - S$, we get all those tuples that should not go to the result in the following expression.

  $$\pi_{R-S}((\pi_{R-S}(r) \times s) - r)$$

- Finally, we subtract this set from the set of all candidate tuples and obtain the output relation of the division operator.

  $$r \div s = \pi_{R-S}(r) - \pi_{R-S}((\pi_{R-S}(r) \times s) - r)$$