Set Operators

24

- Union, Intersection and Difference are defined only for **union compatible** relations.
- Two relations are union compatible if they have the same set of attributes and the types (domains) of the attributes are the same.
- E.g., two relations that are not union compatible:
  - **Student (sNumber, sName)**
  - **Course (cNumber, cName)**

24

Relational Algebra on Bags

25

- A *bag* (or *multiset*) is like a set, but an element may appear more than once.
- Example: {1,2,1,3} is a bag.
- Example: {1,2,3} is also a bag that happens to be a set.

25

Union: ∪

26

- Consider two bags R_1 and R_2 that are union-compatible. Suppose a tuple t appears in R_1 m times, and in R_2 n times. Then in the union, t appears $m + n$ times.

R_1

A	B
1	2
3	4
1	2

R_2

A	B
1	2
3	4
5	6

$R_1 \cup R_2$

A	B
1	2
1	2
1	2
3	4
3	4
5	6

26

Intersection: ∩

27

- Consider two bags R_1 and R_2 that are union-compatible. Suppose a tuple t appears in R_1 m times, and in R_2 n times. Then in the intersection, t appears min (m, n) times.

R_1

A	B
1	2
3	4
1	2

R_2

A	B
1	2
3	4
5	6

$R_1 \cap R_2$

A	B
1	2
3	4

27

Difference: -

28

- Consider two bags R_1 and R_2 that are union-compatible. Suppose a tuple t appears in R_1 m times, and in R_2 n times. Then in $R_1 - R_2$, t appears max (0, m - n) times.

R_1

A	B
1	2
3	4
1	2

R_2

A	B
1	2
3	4
5	6

$R_1 - R_2$

A	B
1	2

28

Building Complex Expressions

29

- Combine operators with parentheses and precedence rules.
- Three notations, just as in arithmetic:
  - Sequences of assignment statements.
  - Expressions with several operators.
  - Expression trees.

Credit: Renee J. Miller

29

Sequences of Assignments

30

- Create temporary relation names.
- Renaming can be implied by giving relations a list of attributes.
  - $\pi_{A+B->C,A->A1,A->A2}$ (R)
- Example: R3 := R1 $\bowtie_C$ R2 can be written:

R4 := R1 X R2

R3 := σ_C (R4)

30

Expressions in a Single Assignment

31

Example: the theta-join R3 := R1 $\bowtie_C$ R2 can be written as

- R3 := σ_C (R1 X R2)

- Precedence of relational operators: (parentheses supercedes)
  - [σ, π, ρ] (highest).
  - [X, $\bowtie$].
  - $\cap$.
  - [$\cup$, —]

31

Expression Trees

32

- Leaves are operands --- either variables standing for relations or particular, constant relations.
- Interior nodes are operators, applied to their child or children.

32

Example: Tree for a Query

33

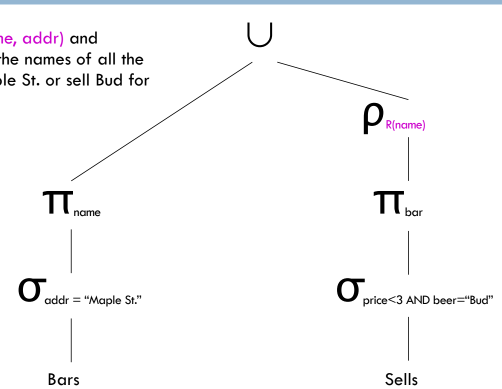
- Using the relations Bars(name, addr) and Sells(bar, beer, price), find the names of all the bars that are either on Maple St. or sell Bud for less than $3.

33

As a Tree:

34

Using the relations Bars(name, addr) and Sells(bar, beer, price), find the names of all the bars that are either on Maple St. or sell Bud for less than $3.



34

Example: Self-Join

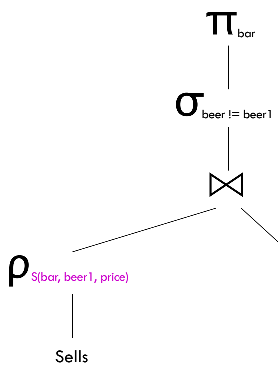
35

- Using Sells(bar, beer, price), find the bars that sell two different beers at the same price.
- Strategy: by renaming, define a copy of Sells, called S(bar, beer1, price). The natural join of Sells and S consists of quadruples (bar, beer, beer1, price) such that the bar sells both beers at this price.

35

The Tree

36



Using Sells(bar, beer, price), find the bars that sell two different beers at the same price.

36

Schemas for Results

37

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

37

Schemas for Results

38

- Product: schema is the attributes of both relations.
  - Distinguish two attributes with the same name.
- Theta-join: same as product.
- Natural join: union of the attributes of the two relations. Keep only one copy of the equated attributes.
- Renaming: the operator tells the schema.

38

Lecture Example

39

39

The Extended Algebra

40

δ = eliminate duplicates from bags.

τ = sort tuples.

γ = grouping and aggregation.

Outerjoin : avoids "dangling tuples" = tuples that do not join with anything.

Credit: Renee J. Miller

40

Duplicate Elimination

41

- R1 := δ(R2).
- R1 consists of one copy of each tuple that appears in R2 one or more times.

41

Example: Duplicate Elimination

42

R = (

A	B)
1	2
3	4
1	2

δ(R) =

A	B
1	2
3	4

42

Sorting

43

- R1 := τ_L (R2).
  - *L* is a list of some of the attributes of R2.
- R1 is the list of tuples of R2 sorted first on the value of the first attribute on *L*, then on the second attribute of *L*, and so on.
  - Break ties arbitrarily.

43

Example: Sorting

44

R = (A B)

A	B
1	2
3	4
5	2

τ_B (R) =

(A B)

A	B
5	2
1	2
3	4

44

Aggregation Operators

45

- Aggregation operators are not formally operators of relational algebra.
- Rather, they apply to entire columns of a table and produce a single result.
- The most important examples: SUM, AVG, COUNT, MIN, and MAX.

45

Example: Aggregation

46

R = (A B)

A	B
1	3
3	4
3	2

SUM(A) = 7
COUNT(A) = 3
MAX(B) = 4
AVG(B) = 3

46

Grouping Operator

47

- R1 := γ_L (R2). *L* is a list of elements that are either:
  1. Individual (*grouping*) attributes.
  2. AGG(*A*), where AGG is one of the aggregation operators and *A* is an attribute.
     - An arrow and a new attribute name renames the component.

47

Applying γ_L(R)

48

- Group *R* according to all the grouping attributes on list *L*.
  - That is: form one group for each distinct list of values for those attributes in *R*.
- Within each group, compute AGG(*A*) for each aggregation on list *L*.
- Result has one tuple for each group:
  1. The grouping attributes and
  2. The group's aggregations.

48

Example: Grouping/Aggregation

49

R =

A	B	C
1	2	3
4	5	6
1	2	5

$\gamma_{A,B,AVG(C)\to X}$ (R) = ??

First, group *R* by *A* and *B* :

A	B	C
1	2	3
1	2	5
4	5	6

Then, average C within groups:

A	B	X
1	2	4
4	5	6

49

Recall: Outerjoin

50

- Suppose we join $R \bowtie_C S$.
- A tuple of *R* that has no tuple of *S* with which it joins is said to be *dangling*.
  - Similarly for a tuple of *S*.
- Outerjoin preserves dangling tuples by padding them NULL.

50

Example: Outerjoin

51

R =

A	B
1	2
4	5

S =

B	C
2	3
6	7

(1,2) joins with (2,3), but the other two tuples are dangling.

R FULL OUTERJOIN S =

A	B	C
1	2	3
4	5	NULL
NULL	6	7

51

Outer Join – Example

52

- instructor

ID	name	dept_name
10101 12121 15151	Srinivasan Wu Mozart	Comp. Sci. Finance Music

teaches

ID	course_id
10101 12121 76766	CS-101 FIN-201 BIO-101

instructor ⋈ *teaches*

ID	name	dept_name	course_id
10101 12121	Srinivasan Wu	Comp. Sci. Finance	CS-101 FIN-201

- Left Outer Join

instructor $⟕_{LEFT}$ *teaches*

ID	name	dept_name	course_id
10101 12121 15151	Srinivasan Wu Mozart	Comp. Sci. Finance Music	CS-101 FIN-201 *NULL*

52

Outer Join – Example

53

- instructor

ID	name	dept_name
10101 12121 15151	Srinivasan Wu Mozart	Comp. Sci. Finance Music

teaches

ID	course_id
10101 12121 76766	CS-101 FIN-201 BIO-101

- *instructor* $⟖_{RIGHT}$ *teaches*

ID	name	dept_name	course_id
10101 12121 76766	Srinivasan Wu null	Comp. Sci. Finance null	CS-101 FIN-201 BIO-101

- *instructor* $⟗_{FULL}$ *teaches*

ID	name	dept_name	course_id
10101 12121 15151 76766	Srinivasan Wu Mozart null	Comp. Sci. Finance Music null	CS-101 FIN-201 *null* BIO-101

53

Operations on Bags

54

A **bag** = a set with repeated elements

All operations need to be defined carefully on bags

- {a,b,b,c}∪{a,b,b,b,e,f,f}={a,a,b,b,b,b,b,c,e,f,f}

- $\sigma_C(R)$: preserve the number of occurrences
- $\Pi_A(R)$: no duplicate elimination
- Cartesian product, join: no duplicate elimination

Important ! Relational Engines work on bags, not sets !

54

Why Bags?

55

- SQL, the most important query language for relational databases, is actually a bag language.
- Some operations, like projection, are more efficient on bags than sets.

55

Operations on Bags

56

- Selection applies to each tuple, so its effect on bags is like its effect on sets.
- Projection also applies to each tuple, we do not eliminate duplicates.
- Products and joins are done on each pair of tuples, so duplicates in bags have no effect on how we operate.

56

Example: Bag Selection

57

R(

A,	B)
1	2
5	6
1	2

$\sigma_{A+B<5}$ (R) =

A	B
1	2
1	2

57

Example: Bag Projection

58

R(

A,	B)
1	2
5	6
1	2

π_A (R) =

A
1
5
1

58

Example: Bag Product

59

R(

A,	B)
1	2
5	6
1	2

S(

B,	C)
3	4
7	8

R X S =

A	R.B	S.B	C
1	2	3	4
1	2	7	8
5	6	3	4
5	6	7	8
1	2	3	4
1	2	7	8

59

Example: Bag Theta-Join

60

R(A, B)

A	B
1	2
5	6
1	2

S(B, C)

B	C
3	4
7	8

$R \bowtie_{R.B<S.B} S =$

A	R.B	S.B	C
1	2	3	4
1	2	7	8
5	6	7	8
1	2	3	4
1	2	7	8

60

Bag Union

61

- An element appears in the union of two bags the sum of the number of times it appears in each bag.
- Example: {1,2,1} ∪ {1,1,2,3,1} = {1,1,1,1,1,2,2,3}

61

Bag Intersection

62

- An element appears in the intersection of two bags the minimum of the number of times it appears in either bag
- Example: {1,2,1,1} ∩ {1,2,1,3} = {1,1,2}.

62

Bag Difference

63

- An element appears in the difference $A - B$ of bags as many times as it appears in *A*, minus the number of times it appears in *B*.
- Example: {1,2,1,1} – {1,2,3} = {1,1}.

63

Beware: Bag Laws != Set Laws

64

- Some, but *not all* algebraic laws that hold for sets also hold for bags.
- Example: the commutative law for union ($R \cup S = S \cup R$) *does* hold for bags.
  - Since addition is commutative, adding the number of times x appears in R and S doesn't depend on the order of R and S.

64

Example: A Law That Fails

65

- Set union is *idempotent*, meaning that $S \cup S = S$.
- However, for bags, if x appears n times in S, then it appears $2n$ times in $S \cup S$.
- Thus $S \cup S$!= S in general.
  - e.g., {1} ∪ {1} = {1,1} != {1}.

What about Intersection?

65

Lecture Example

66

66