RELATIONAL DATABASES
5. Extended relational algebra

Carmelo Vaccaro

University of Paris-Saclay

Master 1 - AI
2022/23: first semester

The content of these slides is taken from the book
Database Systems - The Complete Book,
by Hector Garcia-Molina, Jeffrey D. Ullman and Jennifer Widom,
published by Pearson, 2014.

Why study extended relational algebra

Extended relational algebra is the extension of relational algebra from sets to multisets.

The reason to study extended relational algebra is that commercial DBMS's implement relations that are multisets, rather than sets. Thus extended relational algebra is closer to the real behavior of DBMS's than classical relational algebra.

Contents of the chapter

1. Relational operations on multisets

Multisets

Informally a *multiset* is a set where elements can be repeated. For example $\{1, 2, 1\}$ is a multiset because the element 1 is repeated twice. The difference between a multiset and a list is that in a multiset the order does not count.

The formal definition of multisets is the following. A *multiset* is a pair (U, μ) where U is a set and μ is a function from U to the non-zero natural numbers. The value of μ on an element of U is the *multiplicity* of that element. If $t \in U$, we say that t *appears* $\mu(t)$ *times* in (U, μ).

Union and intersection of multisets

Let $R_1 := (U_1, \mu_1)$ and $R_2 := (U_2, \mu_2)$ be multisets.

The *union* $R_1 \cup R_2$ is the multiset $(U_1 \cup U_2, \mu)$, where $\mu = \mu_1 + \mu_2$ on $U_1 \cap U_2$, $\mu = \mu_1$ on $U_1 \setminus U_2$ and $\mu = \mu_2$ on $U_2 \setminus U_1$. In particular if t is an element appearing n_1 times in R_1 and n_2 times in R_2, then t appears $n_1 + n_2$ times in $R_1 \cup R_2$.

The *intersection* $R_1 \cap R_2$ is the multiset $(U_1 \cap U_2, \mu)$, where $\mu(t) = \min\big(\mu_1(t), \mu_2(t)\big)$.

Difference of multisets

Let $R_1 := (U_1, \mu_1)$ and $R_2 := (U_2, \mu_2)$ be multisets.

Let U be the set of elements $t \in U_1 \cap U_2$ such that $\mu_1(t) > \mu_2(t)$. Then the *difference* $R_1 \setminus R_2$ is the multiset $\big((U_1 \setminus U_2) \cup U, \mu\big)$, where $\mu(t) = \mu_1(t) - \mu_2(t)$ if $t \in U$ and $\mu(t) = \mu_1(t)$ if $t \in U_1 \setminus U_2$.

In particular if t appears n_1 times in R_1 and n_2 times in R_2, then t appears $n_1 - n_2$ times in $R_1 \setminus R_2$.

Example

Let R and S be the following multisets of tuples

Table: R

A	B
1	2
3	4
1	2
1	2

Table: S

A	B
1	2
3	4
3	4
5	6

Then $R \cup S$ is multiset where (1, 2) appears four times (three times for R and once for S); (3, 4) appears three times, and (5, 6) appears once.

Table: $R \cap S$

A	B
1	2
3	4

Table: $R \setminus S$

A	B
1	2
1	2

Projection on multisets of tuples

If R is a multiset of tuples and $A_1, A_2, \cdots, A_n$ are attributes of R, then the projection $\pi_{A_1,A_2,\cdots,A_n}(R)$ is obtained from R by eliminating those columns not in $A_1, A_2, \cdots, A_n$.

Duplicate rows will not be eliminated as for the projection of sets of tuples, thus $\pi_{A_1,A_2,\cdots,A_n}(R)$ will have the same number of rows as R.

Example

If R is the table

A	B	C
1	2	5
3	4	6
1	2	7
1	2	8

then $\pi_{A,B}(R)$ is the table

A	B
1	2
3	4
1	2
1	2

Selection on multisets of tuples

The selection operator applied to multisets of tuples selects tuples based on a condition. Duplicate rows are not eliminated.

Example

If R is the table

A	B	C
1	2	5
3	4	6
1	2	7
1	2	8

then the result of the selection $\sigma_{C \geq 6}(R)$ is

A	B	C
3	4	6
1	2	7
1	2	8

Cartesian product of multisets of tuples

Let R and S be multisets of tuples. Then in the Cartesian product of R with S each tuple of R is paired with each tuple of S, regardless of whether it is a duplicate or not.

As a result, if a tuple r appears m times in R, and the tuple s appears n times in S, then in the Cartesian product $R \times S$, the tuple rs will appear mn times.

Example

Table: R

A	B
1	2
1	2

Table: S

B	C
2	3
4	5
4	5

Table: $R \times S$

A	$R.B$	$S.B$	C
1	2	2	3
1	2	4	5
1	2	4	5
1	2	2	3
1	2	4	5
1	2	4	5

Joins of of multisets of tuples

For doing a join on multisets of tuples we compare each tuple of one relation with each tuple of the other, and if the condition on the join is true we join the pair of tuples.

As usual we do not eliminate duplicate tuples.

Example

Table: R

A	B
1	2
1	2

Table: S

B	C
2	3
4	5
4	5

Table: $R \bowtie S$

A	B	C
1	2	3
1	2	3

Table: R

A	B
1	2
1	2

Table: S

B	C
2	3
4	5
4	5

Table: $R \bowtie_{(R.B<S.B)} S$

A	$R.B$	$S.B$	C
1	2	4	5
1	2	4	5
1	2	4	5
1	2	4	5

2. Extended operators of relational algebra

List of extended operators

In this section we introduce the following extended operators of relational algebra:

1. the duplicate-elimination operator δ;
2. aggregation operators;
3. grouping operator γ;
4. the extended projection;
5. the sorting operator τ;
6. the outerjoin operator.

Short description of extended operators 1/2

1. the duplicate-elimination operator δ turns a multiset into a set by eliminating all but one copy of each tuple.
2. aggregation operators (sums, average, ...) are used by the grouping operator and applied to attributes of a relation.
3. the grouping of tuples according to their value in one or more attributes partitions the tuples of a relation into "groups." Aggregation can then be applied to columns within each group.

Short description of extended operators 2/2

4. the extended projection not only projects out some columns, but can also perform computations to produce new columns.
5. the sorting operator τ turns a relation into a list of tuples, sorted according to one or more attributes.
6. the outerjoin operator is a variant of the join that avoids losing dangling tuples. In the result of the outerjoin, dangling tuples are "padded" with the null value, so the dangling tuples can be represented in the output.

Duplicate elimination

When we need to convert a multiset to a set we use the operator δ, which returns one copy of every tuple that appears one or more times in a relation.

Table: R

A	B
1	2
3	4
1	2
1	2

Table: $\delta(R)$

A	B
1	2
3	4

Exercise

Let R and S be the following tables

Table: R

A	B
0	1
2	3
0	1
2	4
3	4

Table: S

B	C
0	1
2	4
2	5
3	4
0	2
3	4

Compute the following:

a. $\delta(R)$;

b. $\delta(S)$.

Solution

Table: $\delta(R)$

A	B
0	1
2	3
2	4
3	4

Table: $\delta(S)$

B	C
0	1
2	4
2	5
3	4
0	2

Aggregation operators

The aggregation operators summarize or "aggregate" the values in one column of a relation. They are:

1. `SUM` produces the sum of a column with numerical values.
2. `AVG` produces the average of a column with numerical values.
3. `MIN` and `MAX`, applied to a column with numerical values, produce the smallest or largest value, respectively. When applied to a column with character-string values, they produce the lexicographically (alphabetically) first or last value, respectively.
4. `COUNT` produces the number of (not necessarily distinct) values in a column. Equivalently, `COUNT` applied to any attribute of a relation produces the number of tuples of that relation, including duplicates.

Example

Let us consider the relation

Table: R

A	B
1	2
3	4
1	2
1	2

Some examples of aggregations on the attributes of this relation are:

1. $\texttt{SUM}(B) = 2 + 4 + 2 + 2 = 10$.
2. $\texttt{AVG}(A) = (1 + 3 + 1 + 1)/4 = 1.5$.
3. $\texttt{MIN}(A) = 1$.
4. $\texttt{MAX}(B) = 4$.
5. $\texttt{COUNT}(A) = 4$.

Grouping: an example 1/2

Often we do not want simply the average or some other aggregation of an entire column. Rather, we need to consider the tuples of a relation in groups, corresponding to the value of one or more other columns, and we aggregate only within each group.

For example, suppose we wanted to compute the total number of minutes of movies produced by each studio, i.e., a relation such as:

studioName	*sumOfLengths*
Disney	12345
MGM	54321
...	...

Grouping: an example 2/2

Starting with the relation

$$Movies(title, year, length, genre, studioName, producerC\#)$$

we must group the tuples according to their value for attribute *studioName* and then apply the aggregation SUM(*length*) to each group independently.

The grouping operation 1/2

The grouping operator allows to do aggregation within groups.

The grouping operator is denoted γ. Suppose that γ is applied to a relation R; then γ has in the subscript the following information:

1. zero or more *grouping attributes*: the attribute of R by which the R will be separated into groups;
2. a list of one or more *aggregation operations*. Each operations is made by an aggregation operators applied to an attribute of R;
3. optionally an arrow following the aggregation operation and followed by a name, indicating the name of the column generated by the aggregation operation.

The grouping operation 2/2

The relation returned by the grouping operation is constructed as follows:

1. Partition the tuples of R into groups. Each group consists of all the tuples having the same values in the grouping attributes. If there are no grouping attributes, the entire relation R is one group.
2. For each group, produce one tuple consisting of:
   1. the values of the grouping attributes for that group and
   2. the values of the aggregation operations over all the tuples of that group.

Example

Suppose we have the relation *StarsIn(title, year, starName)* and we wish to find, for each star the first year that star has made a movie and also the number of movies made by that star.

Thus we group using *starName* as a grouping attribute and we compute for each group $\texttt{MIN}(year)$ and $\texttt{COUNT}(title)$.

We denote this expression as

$$\gamma_{starName,\texttt{MIN}(year),\texttt{COUNT}(title)}(StarsIn)$$

If we want to rename the attributes of the new table as *minYear* and *ctTitle*, we have to use this notation

$$\gamma_{starName,\texttt{MIN}(year)\rightarrow minYear,\texttt{COUNT}(title)\rightarrow ctTitle}(StarsIn)$$

Exercise

Let R and S be the following tables

Table: R

A	B
0	1
2	3
0	1
2	4
3	4

Table: S

B	C
0	1
2	4
2	5
3	4
0	2
3	4

Compute the following:

a. $\gamma_{A,\texttt{SUM}(B)}(R)$;

b. $\gamma_{B,\texttt{AVG}(B)}(S)$.

Solution

Table: $\gamma_{A,\,\mathtt{SUM}(B)}(R)$

A	$\mathtt{SUM}(B)$
0	2
2	7
3	4

Table: $\gamma_{B,\,\mathtt{AVG}(B)}(S)$

B	$\mathtt{AVG}(B)$
0	0
2	2
3	3

Extended projection

The extended projection is denoted π as the usual projection and the difference with the latter is that the projected attributes can be results of operations on attributes.

Suppose that π is applied to a relation R; then π has in the subscript a list of the following elements in any order:

1. single attributes of R;
2. expressions involving attributes of R, constants, arithmetic operators, and string operators.
3. optionally an arrow following an attribute or an expression, indicating the new name of the column made by the attribute or the expression.

Example

Let R be the relation

A	B	C
0	1	2
0	1	2
3	4	5

Then the result of $\pi_{A,B+C}(R)$ is

A	$B + C$
0	3
0	3
3	9

We can rename X the second column in this way, $\pi_{A,B+C \to X}(R)$

A	X
0	3
0	3
3	9

Exercise

Let R and S be the following tables

Table: R

A	B
0	1
2	3
0	1
2	4
3	4

Table: S

B	C
0	1
2	4
2	5
3	4
0	2
3	4

Compute the following:

a. $\pi_{A+B,\, A^2,\, B^2}(R)$;

b. $\pi_{B+1,\, C-1}(S)$.

Solution

Table: $\pi_{A+B,\, A^2,\, B^2}(R)$

$A + B$	A^2	B^2
1	0	1
5	4	9
1	0	1
6	4	16
7	9	16

Table: $\pi_{B+1,\, C-1}(S)$

$B + 1$	C - 1
1	0
3	3
3	4
4	3
1	1
4	3

The sorting operator

The sorting operator is denoted τ. If R is a relation and $A_1, A_2, \cdots, A_n$ are attributes of R, then the expression $\tau_{A_1,A_2,\cdots,A_n}(R)$ denotes the tuples of R sorted first by their value of A_1.

Ties are broken according to the value of A_2; tuples that agree on both A_1 and A_2 are ordered according to their value of A_3, and so on. Ties that remain after attribute A_n is considered may be ordered arbitrarily.

Example

If R is a relation with schema $R(A, B, C)$, then $\tau_{C,B}(R)$ orders the tuples of R by their value of C, and tuples with the same C-value are ordered by their B value. Tuples that agree on both B and C may be ordered arbitrarily.

Exercise

Let R and S be the following tables

Table: R

A	B
0	1
2	3
0	1
2	4
3	4

Table: S

B	C
0	1
2	4
2	5
3	4
0	2
3	4

Compute the following:

a. $\tau_{B,A}(R)$;

b. $\tau_{B,C}(S)$.

Exercise

Table: $\tau_{B,A}(R)$

A	B
0	1
0	1
2	3
2	4
3	4

Table: $\tau_{B,C}(S)$

B	C
0	1
0	2
2	4
2	5
3	4
3	4

Outerjoins

A property of the join operator is that it is possible for certain tuples to be "dangling"; that is, they fail to match any tuple of the other relation in the common attributes.

Dangling tuples do not have any trace in the result of the join, so the information contained in these tuples is not represented in the join.

The "outerjoin" is a variation on the join that avoids this behavior. Outerjoins exist in various commercial systems.

The natural outerjoin

If R and S are relations, the *natural outerjoin* between R and S, denoted $R \overline{\bowtie} S$ is obtained from the natural join $R \bowtie S$ by adding any dangling tuples from R or S.

The added tuples must be padded with a special null symbol, $\perp$, in all the attributes that they do not possess but that appear in the join result. Note that $\perp$ is written NULL in SQL.

Example

Table: R

A	B	C
1	2	3
4	5	6
7	8	9

Table: S

B	C	D
2	3	10
2	3	11
6	7	12

Table: $R \overline{\bowtie} S$

A	B	C	D
1	2	3	10
1	2	3	11
4	5	6	$\perp$
7	8	9	$\perp$
$\perp$	6	7	12

Tuple (1, 2, 3) of R joins with both (2, 3, 10) and (2, 3, 11) of S, so these three tuples are not dangling. However, the other three tuples, (4, 5, 6) and (7, 8, 9) of R and (6, 7, 12) of S are dangling. The three dangling tuples are padded with $\perp$ in the attributes that they do not have.

Left and right natural outerjoins

The *left outerjoin* $R_{left}\overline{\bowtie}S$ is like the outerjoin, but only dangling tuples of the left argument R are padded with $\perp$ and added to the result.

The *right outerjoin* $R_{right}\overline{\bowtie}S$ is like the outerjoin, but only the dangling tuples of the right argument S are padded with $\perp$ and added to the result.

Example

Table: R

A	B	C
1	2	3
4	5	6
7	8	9

Table: S

B	C	D
2	3	10
2	3	11
6	7	12

Table: $R_{left} \overline{\bowtie} S$

A	B	C	D
1	2	3	10
1	2	3	11
4	5	6	$\perp$
7	8	9	$\perp$

Only the tuples (4, 5, 6) and (7, 8, 9) of R are padded with $\perp$ and figure in the result. The dangling tuple (6, 7, 12) of S is absent.

Example

Table: R

A	B	C
1	2	3
4	5	6
7	8	9

Table: S

B	C	D
2	3	10
2	3	11
6	7	12

Table: $R_{right} \overline{\bowtie} S$

A	B	C	D
1	2	3	10
1	2	3	11
$\perp$	6	7	12

Only the tuple (6, 7, 12) of S is padded with $\perp$ and figure in the result. The dangling tuples (4, 5, 6) and (7, 8, 9) of R are absent.

Theta outerjoins

In addition, all three natural outerjoin operators have theta-join analogs, where first a theta-join is taken and then those tuples that failed to join with any tuple of the other relation, when the condition of the theta-join was applied, are padded with $\perp$ and added to the result.

We use $R \overline{\bowtie}_C S$ to denote a theta outerjoin with condition C. This operator can also be modified to indicate left- or right-outerjoin.

Example

Table: R

A	B	C
1	2	3
4	5	6
7	8	9

Table: S

B	C	D
2	3	10
2	3	11
6	7	12

Table: $R \overline{\bowtie}_{A>S.C} S$

A	$R.B$	$R.C$	$S.B$	$S.C$	D
4	5	6	2	3	10
4	5	6	2	3	11
7	8	9	2	3	10
7	8	9	2	3	11
1	2	3	$\perp$	$\perp$	$\perp$
$\perp$	$\perp$	$\perp$	6	7	12

Exercise

Let R and S be the following tables

Table: R

A	B
0	1
2	3
0	1
2	4
3	4

Table: S

B	C
0	1
2	4
2	5
3	4
0	2
3	4

Compute the following:

a. $R_{left} \overline{\bowtie} S$;

b. $R_{right} \overline{\bowtie} S$;

c. $R \overline{\bowtie} S$;

d. $R \overline{\bowtie}_{R.B<S.B} S$.

Solution 1/2

Table: $R_{left} \overline{\bowtie} S$

A	B	C
0	1	$\perp$
2	3	4
2	3	4
0	1	$\perp$
2	4	$\perp$
3	4	$\perp$

Table: $R_{right} \overline{\bowtie} S$

A	B	C
$\perp$	0	1
$\perp$	2	4
$\perp$	2	5
2	3	4
$\perp$	0	2
2	3	4

Solution 2/2

Table: $R \overline{\bowtie} S$

A	B	C
0	1	$\perp$
2	3	4
2	3	4
0	1	$\perp$
2	4	$\perp$
3	4	$\perp$
$\perp$	0	1
$\perp$	2	4
$\perp$	2	5
$\perp$	0	2

Table: $R \overline{\bowtie}_{R.B<S.B} S$

A	$R.B$	$S.B$	C
0	1	2	4
0	1	2	5
0	1	3	4
0	1	3	4
2	3	$\perp$	$\perp$
0	1	2	4
0	1	2	5
0	1	3	4
0	1	3	4
2	4	$\perp$	$\perp$
3	4	$\perp$	$\perp$
$\perp$	$\perp$	0	1
$\perp$	$\perp$	0	2

The end.