Chapter 5

Algebraic and Logical Query Languages

We now switch our attention from modeling to programming for relational databases. We start in this discussion with two abstract programming languages, one algebraic and the other logic-based. The algebraic programming language, relational algebra, was introduced in Section 2.4, to let us see what operations in the relational model look like. However, there is more to the algebra. In this chapter, we extend the set-based algebra of Section 2.4 to bags, which better reflect the way the relational model is implemented in practice. We also extend the algebra so it can handle several more operations than were described previously; for example, we need to do aggregations (e.g., averages) of columns of a relation.

We close the chapter with another form of query language, based on logic. This language, called "Datalog," allows us to express queries by describing the desired results, rather than by giving an algorithm to compute the results, as relational algebra requires.

5.1 Relational Operations on Bags

In this section, we shall consider relations that are bags (multisets) rather than sets. That is, we shall allow the same tuple to appear more than once in a relation. When relations are bags, there are changes that need to be made to the definition of some relational operations, as we shall see. First, let us look at a simple example of a relation that is a bag but not a set.

Example 5.1: The relation in Fig. 5.1 is a bag of tuples. In it, the tuple $(1, 2)$ appears three times and the tuple $(3, 4)$ appears once. If Fig. 5.1 were a set-valued relation, we would have to eliminate two occurrences of the tuple $(1, 2)$. In a bag-valued relation, we *do* allow multiple occurrences of the same tuple, but like sets, the order of tuples does not matter. □

A	B
1	2
3	4
1	2
1	2

Figure 5.1: A bag

5.1.1 Why Bags?

As we mentioned, commercial DBMS's implement relations that are bags, rather than sets. An important motivation for relations as bags is that some relational operations are considerably more efficient if we use the bag model. For example:

1. To take the union of two relations as bags, we simply copy one relation and add to the copy all the tuples of the other relation. There is no need to eliminate duplicate copies of a tuple that happens to be in both relations.

2. When we project relation as sets, we need to compare each projected tuple with all the other projected tuples, to make sure that each projection appears only once. However, if we can accept a bag as the result, then we simply project each tuple and add it to the result; no comparison with other projected tuples is necessary.

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

Figure 5.2: Bag for Example 5.2

Example 5.2: The bag of Fig. 5.1 could be the result of projecting the relation shown in Fig. 5.2 onto attributes A and B, provided we allow the result to be a bag and do not eliminate the duplicate occurrences of $(1, 2)$. Had we used the ordinary projection operator of relational algebra, and therefore eliminated duplicates, the result would be only:

A	B
1	2
3	4

Note that the bag result, although larger, can be computed more quickly, since there is no need to compare each tuple $(1,2)$ or $(3,4)$ with previously generated tuples. □

Another motivation for relations as bags is that there are some situations where the expected answer can only be obtained if we use bags, at least temporarily. Here is an example.

Example 5.3: Suppose we want to take the average of the A-components of a set-valued relation such as Fig. 5.2. We could not use the set model to think of the relation projected onto attribute A. As a set, the average value of A is 2, because there are only two values of A — 1 and 3 — in Fig. 5.2, and their average is 2. However, if we treat the A-column in Fig. 5.2 as a bag $\{1,3,1,1\}$, we get the correct average of A, which is 1.5, among the four tuples of Fig. 5.2. □

5.1.2 Union, Intersection, and Difference of Bags

These three operations have new definitions for bags. Suppose that R and S are bags, and that tuple t appears n times in R and m times in S. Note that either n or m (or both) can be 0. Then:

- In the bag union $R \cup S$, tuple t appears $n+m$ times.
- In the bag intersection $R \cap S$, tuple t appears $\min(n,m)$ times.
- In the bag difference $R - S$, tuple t appears $\max(0, n-m)$ times. That is, if tuple t appears in R more times than it appears in S, then t appears in $R-S$ the number of times it appears in R, minus the number of times it appears in S. However, if t appears at least as many times in S as it appears in R, then t does not appear at all in $R-S$. Intuitively, occurrences of t in S each "cancel" one occurrence in R.

Example 5.4: Let R be the relation of Fig. 5.1, that is, a bag in which tuple $(1,2)$ appears three times and $(3,4)$ appears once. Let S be the bag

A	B
1	2
3	4
3	4
5	6

Then the bag union $R \cup S$ is the bag in which $(1,2)$ appears four times (three times for its occurrences in R and once for its occurrence in S); $(3,4)$ appears three times, and $(5,6)$ appears once.

The bag intersection $R \cap S$ is the bag

A	B
1	2
3	4

with one occurrence each of $(1,2)$ and $(3,4)$. That is, $(1,2)$ appears three times in R and once in S, and $\min(3,1) = 1$, so $(1,2)$ appears once in $R \cap S$. Similarly, $(3,4)$ appears $\min(1,2) = 1$ time in $R \cap S$. Tuple $(5,6)$, which appears once in S but zero times in R appears $\min(0,1) = 0$ times in $R \cap S$. In this case, the result happens to be a set, but any set is also a bag.

The bag difference $R - S$ is the bag

A	B
1	2
1	2

To see why, notice that $(1,2)$ appears three times in R and once in S, so in $R - S$ it appears $\max(0, 3-1) = 2$ times. Tuple $(3,4)$ appears once in R and twice in S, so in $R - S$ it appears $\max(0, 1-2) = 0$ times. No other tuple appears in R, so there can be no other tuples in $R - S$.

As another example, the bag difference $S - R$ is the bag

A	B
3	4
5	6

Tuple $(3,4)$ appears once because that is the number of times it appears in S minus the number of times it appears in R. Tuple $(5,6)$ appears once in $S - R$ for the same reason. □

5.1.3 Projection of Bags

We have already illustrated the projection of bags. As we saw in Example 5.2, each tuple is processed independently during the projection. If R is the bag of Fig. 5.2 and we compute the bag-projection $\pi_{A,B}(R)$, then we get the bag of Fig. 5.1.

If the elimination of one or more attributes during the projection causes the same tuple to be created from several tuples, these duplicate tuples are not eliminated from the result of a bag-projection. Thus, the three tuples $(1,2,5)$, $(1,2,7)$, and $(1,2,8)$ of the relation R from Fig. 5.2 each gave rise to the same tuple $(1,2)$ after projection onto attributes A and B. In the bag result, there are three occurrences of tuple $(1,2)$, while in the set-projection, this tuple appears only once.

Bag Operations on Sets

Imagine we have two sets R and S. Every set may be thought of as a bag; the bag just happens to have at most one occurrence of any tuple. Suppose we intersect $R \cap S$, but we think of R and S as bags and use the bag intersection rule. Then we get the same result as we would get if we thought of R and S as sets. That is, thinking of R and S as bags, a tuple t is in $R \cap S$ the minimum of the number of times it is in R and S. Since R and S are sets, t can be in each only 0 or 1 times. Whether we use the bag or set intersection rules, we find that t can appear at most once in $R \cap S$, and it appears once exactly when it is in both R and S. Similarly, if we use the bag difference rule to compute $R - S$ or $S - R$ we get exactly the same result as if we used the set rule.

However, union behaves differently, depending on whether we think of R and S as sets or bags. If we use the bag rule to compute $R \cup S$, then the result may not be a set, even if R and S are sets. In particular, if tuple t appears in both R and S, then t appears twice in $R \cup S$ if we use the bag rule for union. But if we use the set rule then t appears only once in $R \cup S$.

5.1.4 Selection on Bags

To apply a selection to a bag, we apply the selection condition to each tuple independently. As always with bags, we do not eliminate duplicate tuples in the result.

Example 5.5: If R is the bag

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

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

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

That is, all but the first tuple meets the selection condition. The last two tuples, which are duplicates in R, are each included in the result. □

Algebraic Laws for Bags

An algebraic law is an equivalence between two expressions of relational algebra whose arguments are variables standing for relations. The equivalence asserts that no matter what relations we substitute for these variables, the two expressions define the same relation. An example of a well-known law is the commutative law for union: $R \cup S = S \cup R$. This law happens to hold whether we regard relation-variables R and S as standing for sets or bags. However, there are a number of other laws that hold when relational algebra is applied to sets but that do not hold when relations are interpreted as bags. A simple example of such a law is the distributive law of set difference over union, $(R \cup S) - T = (R - T) \cup (S - T)$. This law holds for sets but not for bags. To see why it fails for bags, suppose R, S, and T each have one copy of tuple t. Then the expression on the left has one t, while the expression on the right has none. As sets, neither would have t. Some exploration of algebraic laws for bags appears in Exercises 5.1.4 and 5.1.5.

5.1.5 Product of Bags

The rule for the Cartesian product of bags is the expected one. Each tuple of one relation is paired with each tuple of the other, regardless of whether it is a duplicate or not. As a result, if a tuple r appears in a relation R m times, and tuple s appears n times in relation S, then in the product $R \times S$, the tuple rs will appear mn times.

Example 5.6: Let R and S be the bags shown in Fig. 5.3. Then the product $R \times S$ consists of six tuples, as shown in Fig. 5.3(c). Note that the usual convention regarding attribute names that we developed for set-relations applies equally well to bags. Thus, the attribute B, which belongs to both relations R and S, appears twice in the product, each time prefixed by one of the relation names. □

5.1.6 Joins of Bags

Joining bags presents no surprises. We compare each tuple of one relation with each tuple of the other, decide whether or not this pair of tuples joins successfully, and if so we put the resulting tuple in the answer. When constructing the answer, we do not eliminate duplicate tuples.

Example 5.7: The natural join $R \bowtie S$ of the relations R and S seen in Fig. 5.3 is

A	B
1	2
1	2

(a) The relation R

B	C
2	3
4	5
4	5

(b) The relation S

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

(c) The product $R \times S$

Figure 5.3: Computing the product of bags

A	B	C
1	2	3
1	2	3

That is, tuple $(1, 2)$ of R joins with $(2, 3)$ of S. Since there are two copies of $(1, 2)$ in R and one copy of $(2, 3)$ in S, there are two pairs of tuples that join to give the tuple $(1, 2, 3)$. No other tuples from R and S join successfully.

As another example on the same relations R and S, the theta-join

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

produces the bag

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

The computation of the join is as follows. Tuple $(1,2)$ from R and $(4,5)$ from S meet the join condition. Since each appears twice in its relation, the number of times the joined tuple appears in the result is 2×2 or 4. The other possible join of tuples — $(1,2)$ from R with $(2,3)$ from S — fails to meet the join condition, so this combination does not appear in the result. □

5.1.7 Exercises for Section 5.1

Exercise 5.1.1: Let PC be the relation of Fig. 2.21(a), and suppose we compute the projection $\pi_{speed}(\text{PC})$. What is the value of this expression as a set? As a bag? What is the average value of tuples in this projection, when treated as a set? As a bag?

Exercise 5.1.2: Repeat Exercise 5.1.1 for the projection $\pi_{hd}(\text{PC})$.

Exercise 5.1.3: This exercise refers to the "battleship" relations of Exercise 2.4.3.

a) The expression $\pi_{bore}(\texttt{Classes})$ yields a single-column relation with the bores of the various classes. For the data of Exercise 2.4.3, what is this relation as a set? As a bag?

! b) Write an expression of relational algebra to give the bores of the ships (not the classes). Your expression must make sense for bags; that is, the number of times a value b appears must be the number of ships that have bore b.

! **Exercise 5.1.4:** Certain algebraic laws for relations as sets also hold for relations as bags. Explain why each of the laws below hold for bags as well as sets.

a) The associative law for union: $(R \cup S) \cup T = R \cup (S \cup T)$.

b) The associative law for intersection: $(R \cap S) \cap T = R \cap (S \cap T)$.

c) The associative law for natural join: $(R \bowtie S) \bowtie T = R \bowtie (S \bowtie T)$.

d) The commutative law for union: $(R \cup S) = (S \cup R)$.

e) The commutative law for intersection: $(R \cap S) = (S \cap R)$.

f) The commutative law for natural join: $(R \bowtie S) = (S \bowtie R)$.

g) $\pi_L(R \cup S) = \pi_L(R) \cup \pi_L(S)$. Here, L is an arbitrary list of attributes.

h) The distributive law of union over intersection:

$$R \cup (S \cap T) = (R \cup S) \cap (R \cup T)$$