CHAPTER 9 Set Operators

Oracle SQL By Example, Fourth Edition by Alice Rischert. Published by Prentice Hall. Copyright © 2008 by Pearson Education, Inc.

CHAPTER OBJECTIVES

In this chapter, you will learn about:

- The Power of UNION and UNION ALL
- The MINUS and INTERSECT Set Operators

Set operators combine two or more sets of data to produce a single result set. Oracle has four set operators: UNION, UNION ALL, MINUS, and INTERSECT. The UNION and UNION ALL operators combine results. The INTERSECT operator determines common rows. The MINUS operator shows differences between sets of rows.

The sets of data in a set operation are SELECT statements and when writing any set operation, there are two rules to remember.

- Each of the SELECT lists must contain the same number of columns.
- The matching columns in each of the SELECT lists must be the same data type. (Oracle considers

CHAR and VARCHAR2 to be data type compatible.)

In this chapter, you will use set operators to retrieve data from many tables throughout the STUDENT schema.

LAB 9.1 The Power of UNION and UNION ALL

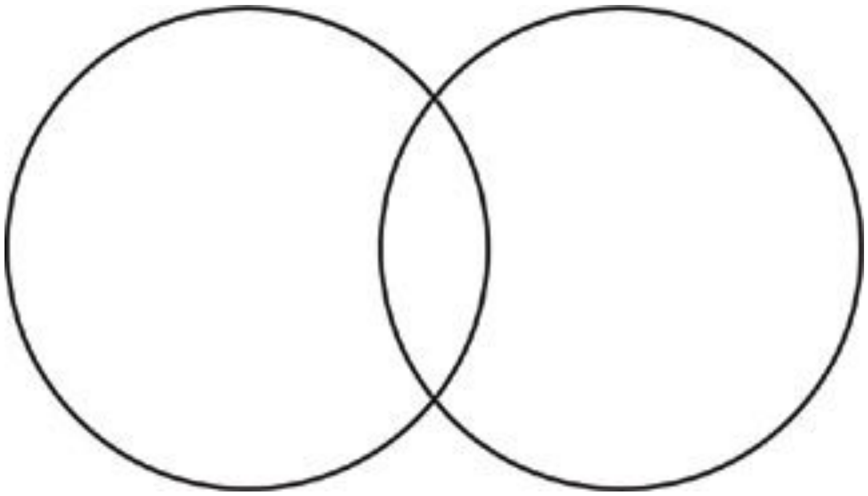
LAB OBJECTIVES

After this lab, you will be able to:

- Use the UNION Set Operator
- Use the UNION ALL Set Operator

The UNION operator is probably the most commonly used set operator. It combines two or more sets of data to produce a single set of data. Think of the UNION operator as two overlapping circles, as illustrated in Figure 9.1. The union of the two circles is everything from both circles. There are duplicates where they overlap, and there may even be duplicates within each set, but the final result shows these values only once. The UNION ALL operator includes these duplicates.

FIGURE 9.1 UNION and UNION ALL set operators



The UNION Operator

Imagine that you need to create a phone list of all instructors and students. The following set operation uses the UNION operator to combine instructor and student names and phone numbers from the INSTRUCTOR and STUDENT tables into a single result set.

```
FIRST_NAME LAST_NAME        PHONE
---------- ---------------- ------------
A.         Tucker           203-555-5555
Adele      Rothstein        718-555-5555

...
Z.A.       Scrittorale      203-555-5555
Zalman     Draquez          718-555-5555

276 rows selected.
```

The same three columns are selected from each table, effectively stacking the columns one on top of the other in the result set. The results are automatically sorted by the order in which the columns appear in the SELECT list.

The result returns 276 rows, even though there are 268 student rows and 10 instructor rows. What happened to the other 2 rows? The following query shows duplicate rows in the STUDENT table.

```
FIRST_NAME LAST_NAME       PHONE          COUNT(*)
---------- --------------- ----------- -----------
Kevin      Porch           201-555-5555          2
Thomas     Edwards         201-555-5555          2

2 rows selected.
```

Because the UNION operator shows only distinct rows, the duplicate student row appears just once in the result of the UNION set operation. To list all the instructors and students, including duplicates, you can take either of two approaches. One approach is to add the ID of the INSTRUCTOR and STUDENT tables to the set operation, plus a text literal such as 'instructor' or 'student'. The other approach is to use the UNION ALL operator.

The UNION ALL Operator

UNION ALL includes any duplicates when sets of data are added. Think again of the two overlapping circles shown in Figure 9.1. UNION ALL not only adds the two sets of data but includes the overlapping duplicates as well. Duplicates that may exist within each set are also included.

```
FIRST_NAME LAST_NAME        PHONE
---------- ---------------- ------------
Fernand    Hanks            2125551212
Tom        Wojick           2125551212
...
Kathleen   Mastandora       718-555-5555
Angela     Torres           718-555-5555

278 rows selected.
```

UNION ALL results in 278 rows, which includes the duplicates in the STUDENT table. Also, the result set is no longer sorted; UNION ALL does not perform a sort. Therefore, a query containing the UNION operator is more time-consuming to execute than a query with the UNION ALL operator. Unless you have a reason to show only distinct rows, use UNION ALL instead of UNION because it will yield better performance.

ORDER BY and Set Operations

Just like the result of any SELECT statement, the result of a set operation can be sorted using the ORDER BY clause. Instead of naming the column by which you want to sort the result, refer to its position in the SELECT list instead. Consider what happens if you add the instructor and student IDs to the previous example by using UNION and order the results by the LAST_NAME column.

```
SELECT instructor_id id, first_name, last_name, phone
  FROM instructor
 UNION
SELECT student_id, first_name, last_name, phone
  FROM student
 ORDER BY 3
       ID FIRST_NAME LAST_NAME        PHONE
--------- ---------- ---------------- ------------
      119 Mardig     Abdou            718-555-5555
      399 Jerry      Abdou            718-555-5555
...
      184 Salewa     Zuckerberg       718-555-5555
      206 Freedon    annunziato       718-555-5555

278 rows selected.
```

The ORDER BY clause can also refer to a column alias, such as the ID used for the first column. However, referring to the column position in the ORDER BY clause is ANSI standard and is also independent of the column names in either SELECT statement.

With the addition of the instructor and student IDs, the unique combination of those IDs with first name, last name, and phone number now produces all 278 rows between the INSTRUCTOR and STUDENT tables.

The first columns in each of the individual SELECT statements, INSTRUCTOR_ID and STUDENT_ID, have different names but are of the same data type. Use an alias to give the column in the result set a meaningful name for both instructor and student IDs.

SQL always takes its cue from the topmost SELECT statement when naming columns in the result set. When you want the result set to display a specific column name that is not dependent on the names of columns listed in the topmost statement, you must use a column alias.

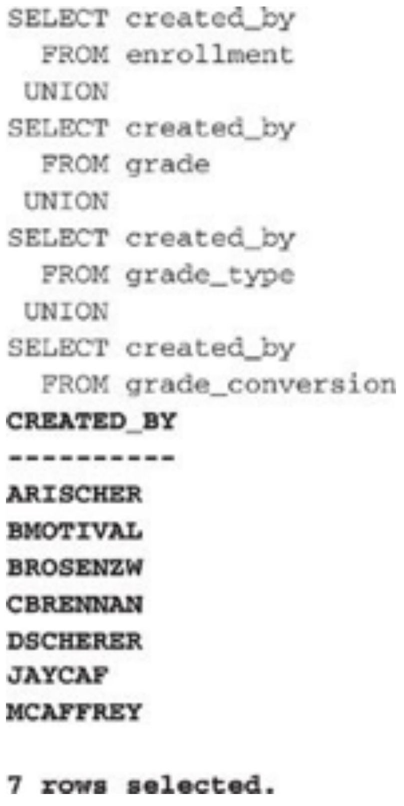
LAB 9.1 EXERCISES

a) Explain the result of the following set operation and explain why it works.

```
SELECT first_name, last_name,
       'Instructor' ”Type”
  FROM instructor
 UNION ALL
SELECT first_name, last_name,
       'Student'
  FROM student
```

b) Write a set operation, using the UNION set operator, to list all the zip codes in the INSTRUCTOR and STUDENT tables.

c) What problem does the following set operation solve?



```
SELECT created_by
  FROM enrollment
 UNION
SELECT created_by
  FROM grade
 UNION
SELECT created_by
  FROM grade_type
 UNION
SELECT created_by
  FROM grade_conversion
CREATED_BY
----------
ARISCHER
BMOTIVAL
BROSENZW
CBRENNAN
DSCHERER
JAYCAF
MCAFFREY

7 rows selected.
```

d) Explain why the result of the following set operation returns an error.

```
SELECT course_no, description
  FROM course
 WHERE prerequisite IS NOT NULL
 ORDER BY 1
 UNION
SELECT course_no, description
  FROM course
 WHERE prerequisite IS NULL
```

e) What is wrong with the following set operation, and what do you have to change to make it work correctly?

```
SELECT instructor_id, last_name
  FROM instructor
 UNION
SELECT last_name, student_id
  FROM student
```

LAB 9.1 EXERCISE ANSWERS

a) Explain the result of the following set operation and explain why it works.

```
SELECT first_name, last_name,
       'Instructor' "Type"
  FROM instructor
 UNION ALL
SELECT first_name, last_name,
       'Student'
  FROM student
```

ANSWER: The result set displays the first and last names of instructors and students. The third column identifies what type of person each is.'Instructor' and 'Student' are both text literals and are in the same position in each SELECT list. Therefore, the two SELECT statements are row compatible.

```
FIRST_NAME LAST_NAME                 Type
---------- ------------------------- -------
A.         Tucker                    Student
Adele      Rothstein                 Student
...
Z.A.       Scrittorale               Student
Zalman     Draquez                   Student

278 rows selected.
```

As your SELECT statements and set operations become more complex, it can be difficult to identify the data in your result sets accurately.

This technique of identifying each row in the result set coming from one or the other set of data may be very useful.

b) Write a set operation, using the UNION set operator, to list all the zip codes in the INSTRUCTOR and STUDENT tables.

ANSWER: The query combines two SELECT statements, using the UNION set operator, for a result set displaying zip codes from both tables, eliminating any duplicates.

```
SELECT zip
  FROM instructor
 UNION
SELECT zip
  FROM student
ZIP
-----
01247
02124
...
43224
48104

149 rows selected.
```

c) What problem does the following set operation solve?

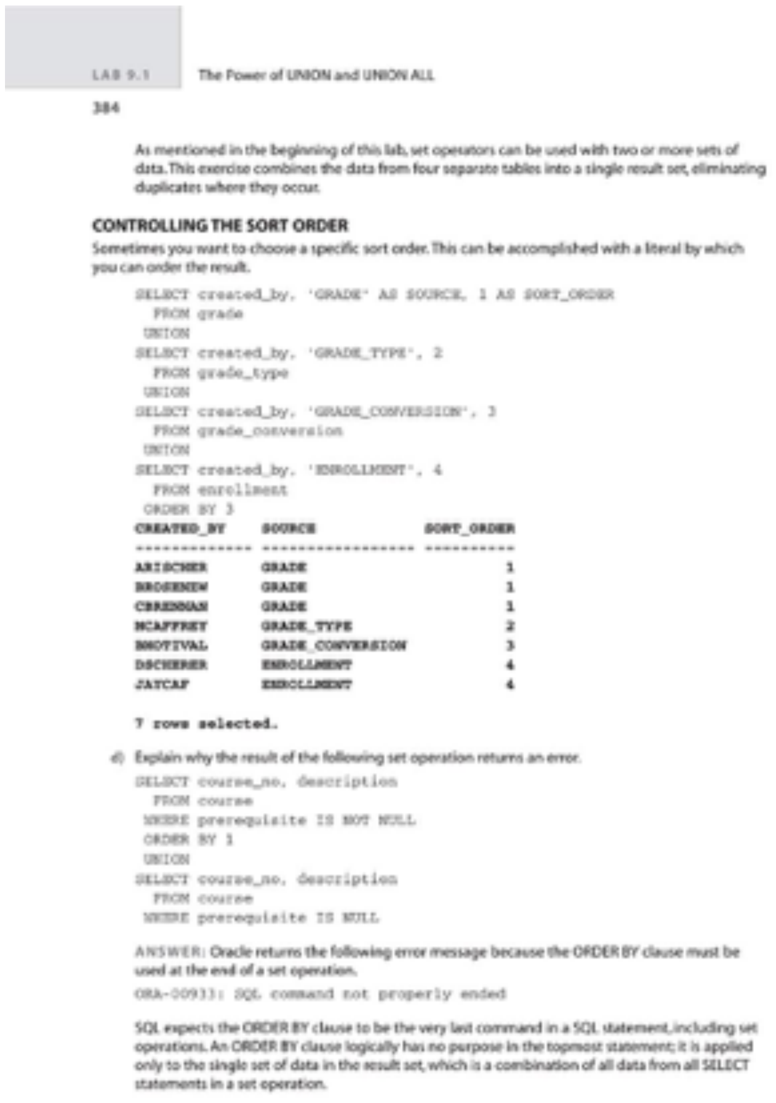
```
SELECT created_by
  FROM enrollment
 UNION
SELECT created_by
  FROM grade
 UNION
SELECT created_by
  FROM grade_type
 UNION
SELECT created_by
  FROM grade_conversion
CREATED_BY
----------
ARISCHER
BMOTIVAL
BROSENZW
CBRENNAN
DSCHERER
JAYCAF
MCAFFREY

7 rows selected.
```

ANSWER: The query displays a list of users who created rows in the ENROLLMENT, GRADE, GRADE_TYPE, and GRADE_CONVERSION tables. It shows each user name only once.

As mentioned in the beginning of this lab, set operators can be used with two or more sets of data. This exercise combines the data from four separate tables into a single result set, eliminating duplicates where they occur.

CONTROLLING THE SORT ORDER

Sometimes you want to choose a specific sort order. This can be accomplished with a literal by which you can order the result.



As mentioned in the beginning of this lab, set operators can be used with two or more sets of data. This exercise combines the data from four separate tables into a single result set, eliminating duplicates where they occur.

CONTROLLING THE SORT ORDER

Sometimes you want to choose a specific sort order. This can be accomplished with a literal by which you can order the result.

```
SELECT created_by, 'GRADE' AS SOURCE, 1 AS SORT_ORDER
  FROM grade
 UNION
SELECT created_by, 'GRADE_TYPE', 2
  FROM grade_type
 UNION
SELECT created_by, 'GRADE_CONVERSION', 3
  FROM grade_conversion
 UNION
SELECT created_by, 'ENROLLMENT', 4
  FROM enrollment
 ORDER BY 3
CREATED_BY    SOURCE            SORT_ORDER
------------- ----------------- ----------
ARISCHER      GRADE                      1
BROSENZW      GRADE                      1
CBRENNAN      GRADE                      1
MCAFFREY      GRADE_TYPE                 2
BMOTIVAL      GRADE_CONVERSION           3
DSCHERER      ENROLLMENT                 4
JAYCAF        ENROLLMENT                 4

7 rows selected.
```

d) Explain why the result of the following set operation returns an error.

```
SELECT course_no, description
  FROM course
 WHERE prerequisite IS NOT NULL
 ORDER BY 1
 UNION
SELECT course_no, description
  FROM course
 WHERE prerequisite IS NULL
```

ANSWER: Oracle returns the following error message because the ORDER BY clause must be used at the end of a set operation.

```
ORA-00933: SQL command not properly ended
```

SQL expects the ORDER BY clause to be the very last command in a SQL statement, including set operations. An ORDER BY clause logically has no purpose in the topmost statement; it is applied only to the single set of data in the result set, which is a combination of all data from all SELECT statements in a set operation.

d) Explain why the result of the following set operation returns an error.

```
SELECT course_no, description
  FROM course
 WHERE prerequisite IS NOT NULL
 ORDER BY 1
 UNION
SELECT course_no, description
  FROM course
 WHERE prerequisite IS NULL
```

ANSWER: Oracle returns the following error message because the ORDER BY clause must be used at the end of a set operation.

ORA-00933: SQL command not properly ended

SQL expects the ORDER BY clause to be the very last command in a SQL statement, including set operations. An ORDER BY clause logically has no purpose in the topmost statement; it is applied only to the single set of data in the result set, which is a combination of all data from all SELECT statements in a set operation.

e) What is wrong with the following set operation, and what do you have to change to make it work correctly?

```
SELECT instructor_id, last_name
  FROM instructor
 UNION
SELECT last_name, student_id
  FROM student
```

ANSWER: Oracle returns the error ORA-01790: expression must have same data type as corresponding expression. The data types of columns must be the same for columns in the same position in each SELECT list of a set operation. Either the order of the columns in the first or the second statement must be switched for the statement to work correctly.

Sometimes, the data types of columns do not match because of the way the columns were created, in which case you can use the data conversion functions to change from one data type to another.

DATA TYPE CONVERSIONS AND NULLS

You want to combine distinct result sets together by using the UNION ALL operator and place a null value for those columns where you want to omit the value. The following query uses the CAST function and the TO_DATE function to make sure the null columns are set to the same data type, and it avoids implicit data type conversion.

```
SELECT DISTINCT salutation, CAST(NULL AS NUMBER),
       state, z.created_date
  FROM instructor i, zipcode z
 WHERE i.zip = z.zip
UNION ALL
SELECT salutation, COUNT(*),
       state, TO_DATE(NULL)
  FROM student s, zipcode z
 WHERE s.zip = z.zip
 GROUP BY salutation, state
SALUT CAST(NULLASNUMBER) ST CREATED_D
----- ------------------ -- ---------
DR                       NY 03-AUG-03
HON                      NY 03-AUG-03
...
MS.                   69 NY
MS.                    1 WV
REV                    1 NJ

19 rows selected.
```

Lab 9.1 Quiz

In order to test your progress, you should be able to answer the following questions.

1) It is redundant to use DISTINCT in a UNION set operation.

______**a)** True

______**b)** False

2) Each of the SELECT statements in a set operation must have an ORDER BY clause when you want the results to be ordered.

______**a)** True

______**b)** False

3) A UNION set operation always returns the same result set as an equijoin.

______**a)** True

______**b)** False

4) You cannot use UNION to combine data from two tables that do not have a primary key/foreign key relationship.

______**a)** True

______**b)** False

5) There must be the same number of columns in each SELECT statement of a set operation.

______**a)** True

______**b)** False

ANSWERS APPEAR IN APPENDIX A.

LAB 9.2 The MINUS and INTERSECT Set Operators

LAB OBJECTIVES

After this lab, you will be able to:

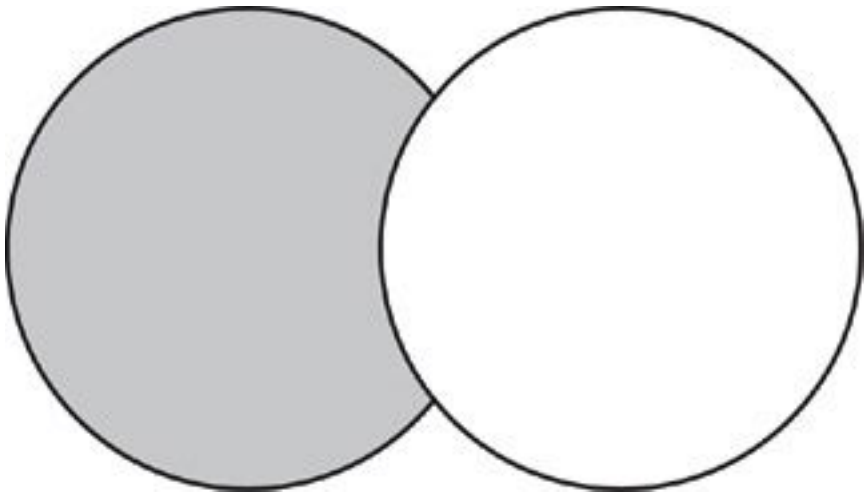
- Use the MINUS Set Operator
- Use the INTERSECT Set Operator

The MINUS set operator subtracts one set of data from another, identifying what data exists in one table but not the other. The INTERSECT set operator is the intersection of sets of data, identifying data common to all of them.

The MINUS Operator

The MINUS operator returns the difference between two sets. Effectively, you use it to subtract one set from another set. The gray area of the circle in Figure 9.2 shows the difference between the sets and indicates the data that is in one circle but not in another.

FIGURE 9.2 The MINUS set operators



The following set operation lists instructors not currently teaching any classes (sections).

```
SELECT instructor_id
  FROM instructor
 MINUS
SELECT instructor_id
  FROM section
INSTRUCTOR_ID
-------------
          109
          110

2 rows selected.
```

The first SELECT statement returns the complete list of instructors.

```
SELECT instructor_id
  FROM instructor
INSTRUCTOR_ID
-------------
          101
          102
          103
          104
          105
          106
          109
          108
          107
          110

10 rows selected.
```

The second SELECT statement returns a distinct list of instructors currently teaching.

```
SELECT DISTINCT instructor_id
  FROM section
INSTRUCTOR_ID
-------------
          101
          102
          103
          104
          105
          106
          107
          108

8 rows selected.
```

Subtracting the second result set from the first result set leaves a list of instructors not currently teaching, which are the INSTRUCTOR_ID values 109 and 110. Just like the UNION set operator, MINUS eliminates duplicates when evaluating sets of data. DISTINCT is used in the preceding second SELECT statement when it is written separately.

The following set operation implies distinct values in both SELECT statements.

```
SELECT created_by
  FROM enrollment
 MINUS
SELECT created_by
  FROM course
CREATED_BY
---------------
JAYCAF

1 row selected.
```

Written separately, the two SELECT statements use DISTINCT.

```
SELECT DISTINCT created_by
  FROM enrollment
CREATED_BY
---------------
DSCHERER
JAYCAF

2 rows selected.
SELECT DISTINCT created_by
  FROM course
CREATED_BY
---------------
DSCHERER

1 row selected.
```

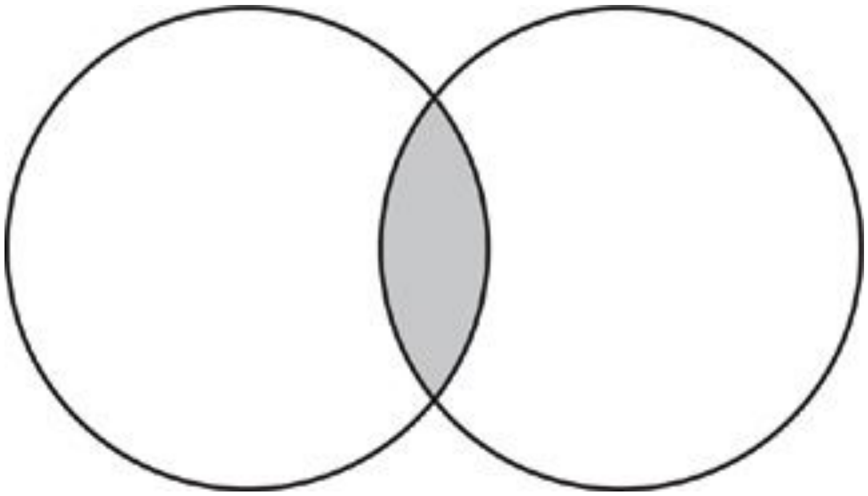
The second SELECT statement results in the distinct value DSCHERER. This is subtracted from the result of the first statement, which consists of the distinct values JAYCAF and DSCHERER. This results in the value JAYCAF because JAYCAF is not found in the COURSE table, only in the ENROLLMENT table. This type of statement, whereby you retrieve data that exists in one table but not in another, is sometimes referred to as an *antijoin*.

Be careful when positioning the SELECT statements in a MINUS set operation because their order makes a big difference. Be sure to place the set you want to subtract from first.

The INTERSECT Operator

The INTERSECT operator determines the common values between two sets. Figure 9.3 illustrates the two overlapping circles, and the gray color indicates the area where the two circles intersect.

FIGURE 9.3 The INTERSECT set operator



When you use INTERSECT instead of MINUS in the previous statement, the result is quite different.

```
SELECT created_by
  FROM enrollment
INTERSECT
SELECT created_by
  FROM course
CREATED_BY
---------------
DSCHERER

1 row selected.
```

The result set contains DSCHERER, which is the distinct value where the two sets overlap or intersect. Unlike with MINUS, the order of the SELECT statements in an INTERSECT set operation does not matter.

USING INTERSECT INSTEAD OF EQUIJOINS

The equijoin discussed in Chapter 7, "Equijoins," produces a result set that is the intersection of two or more tables—the same result as with INTERSECT.

The following is an equijoin that returns a list of course numbers for courses with corresponding sections.

```
SELECT DISTINCT c.course_no
  FROM course c, section s
 WHERE c.course_no = s.course_no

 COURSE_NO
 ---------
        10
        20
 ...
       420
       450

 28 rows selected.
```

This INTERSECT set operation returns the same result.

```
SELECT course_no
  FROM course
INTERSECT
SELECT course_no
  FROM section
COURSE_NO
---------
       10
       20
...
      420
      450

28 rows selected.
```

The drawback to using INTERSECT instead of an equijoin is that INTERSECT operates on all columns in each SELECT list of the set operation. Therefore, you cannot include columns that exist in one table and not the other.

Execution Order of Set Operations

The execution order for all set operations is from top to bottom. You can see the effect in the following query, which involves three tables: T1, T2, and T3. The query combines the values of table T1 and T2, and the subsequent MINUS operation compares this result to the values of T3. (These three tables are not found in the STUDENT schema unless you installed the additional tables from the companion Web site.)

```
SELECT col1
  FROM t1
UNION ALL
SELECT col2
  FROM t2
MINUS
SELECT col3
  FROM t3
      COL1
----------
         1
         4

2 rows selected.
```

You use parentheses to indicate a change in the execution order, as in the following example. The query in parentheses is executed first; it determines the differences between T2 and T3 and then uses the UNION ALL operator to combine the result with T1. The output is quite different from the previous result.

```
SELECT col1
  FROM t1
UNION ALL
(SELECT col2
  FROM t2
MINUS
SELECT col3
  FROM t3)
      COL1
----------
         1
         2
         3
         4

4 rows selected.
```

Comparing Two Tables

Set operators are very useful if you need to determine the differences between two tables. For example, say that you want to compare test and production database tables or check data generated by a program to a previous state of the table. The following statement involves two tables: OLD_TABLE contains the original state of the table, and NEW_TABLE shows the data after the execution of various data manipulation statements. The queries inside the parentheses are executed first, and then the UNION ALL operation is performed. The SQL statement returns the differences between the two tables.

```
(SELECT *
   FROM old_table
MINUS
SELECT *
   FROM new_table)
UNION ALL
(SELECT *
   FROM new_table
MINUS
SELECT *
   FROM old_table)
```

Ideally, your tables have a primary key or unique key to uniquely identify the rows. If that's not the case, duplicate rows in the table may be possible. You might want to include the COUNT(*) function along with all the columns and a GROUP BY clause.

Another way to find differences is to use Oracle's Flashback query, which allows you to list the changes performed between specific time intervals. You will learn more about this feature in Chapter 11, "Insert, Update, and Delete."

LAB 9.2 EXERCISES

a) Explain the result of the following set operation.

```
SELECT course_no, description
  FROM course
 MINUS
SELECT s.course_no, c.description
  FROM section s, course c
 WHERE s.course_no =
       c.course_no
```

b) Use the MINUS set operator to create a list of courses and sections with no students enrolled. Add to the result set a column with the title Status and display the text No Enrollments in each row. Order the results by the COURSE_NO and SECTION_NO columns.

c) Use the appropriate set operator to list all zip codes that are in both the STUDENT and INSTRUCTOR tables.

d) Use the INTERSECT set operator to list student IDs for students who are enrolled.

LAB 9.2 EXERCISE ANSWERS

a) Explain the result of the following set operation.

```
SELECT course_no, description
  FROM course
 MINUS
SELECT s.course_no, c.description
  FROM section s, course c
 WHERE s.course_no =
       c.course_no
```

ANSWER: The set operation subtracts all courses having sections from all courses, resulting in the two courses without matching sections.

```
COURSE_NO DESCRIPTION
--------- --------------------------------
       80 Programming Techniques
      430 Java Developer III

2 rows selected.
```

Another way to formulate the query is to write a subquery using the NOT IN operator or the NOT EXISTS operator.

```
SELECT course_no, description
```

```
FROM course c
WHERE NOT EXISTS
   (SELECT '*'
   FROM section
   WHERE c.course_no = course_no)
```

In Chapter 10, “Complex Joins,” you will learn about using an outer join as another way to achieve a similar result.

b) Use the MINUS set operator to create a list of courses and sections with no students enrolled. Add to the result set a column with the title Status and display the text No Enrollments in each row. Order the results by the COURSE_NO and SECTION_NO columns.

ANSWER: The first SELECT statement is the set of all courses with sections. The second SELECT statement subtracts the set of courses and sections having enrollments, leaving the courses and sections without enrollments.

```
COURSE_NO SECTION_NO Status
--------- ---------- --------------
       25          9 No Enrollments
      124          4 No Enrollments
...
      220          1 No Enrollments
      350          3 No Enrollments

14 rows selected.
```

This statement makes use of the literal 'No Enrollments' in the SELECT statement. Even though it is not a column in either table, as long as it is in the first statement, there is a column for it in the result set. And, as long as it is in the second statement, it matches the first and, therefore, allows the MINUS to work correctly, subtracting one set from a similar set.

c) Use the appropriate set operator to list all zip codes that are in both the STUDENT and INSTRUCTOR tables.

ANSWER: INTERSECT is used to find the intersection of distinct zip codes in the INSTRUCTOR and STUDENT tables.

```
SELECT zip
  FROM instructor
INTERSECT
SELECT zip
  FROM student
  ZIP
  -----
  10025

  1 row selected.
```

Be careful when deciding to use INTERSECT versus UNION. The key phrase in this exercise is "zip codes that are in both." INTERSECT achieves the intersection of both tables alone, whereas UNION returns all zip codes from both tables combined.

d) Use the INTERSECT set operator to list student IDs for students who are enrolled.

ANSWER: The intersection of student IDs in the STUDENT and ENROLLMENT tables yields all students who are enrolled.

```
SELECT student_id
  FROM student
INTERSECT
SELECT student_id
  FROM enrollment
STUDENT_ID
----------
       102
       103
...
       282
       283

165 rows selected.
```

Lab 9.2 Quiz

In order to test your progress, you should be able to answer the following questions.

1) The following two SELECT statements are equivalent and return the same rows.

```
SELECT student_id  SELECT student_id
  FROM enrollment  FROM student
            MINUS  MINUS
SELECT student_id  SELECT student_id
     FROM student  FROM enrollment
```

______**a)** True

______**b)** False

2) The SELECT statements in an INTERSECT set operation can contain a correlated subquery.

______**a)** True

______**b)** False

3) The following SQL statement executes without an error.

```
SELECT TO_CHAR(1)
FROM dual
MINUS
SELECT TO_NUMBER('1')
FROM dual
```

______**a)** True

______**b)** False

4) It is redundant to use DISTINCT in either a MINUS or INTERSECT set operation.

______**a)** True

______**b)** False

ANSWERS APPEAR IN APPENDIX A.

WORKSHOP

The projects in this section are meant to prompt you to utilize all the skills you have acquired throughout this chapter. The answers to these projects can be found at the companion Web site to this book, located at www.oraclesqlbyexample.com.

1) List all the zip codes in the ZIPCODE table that are not used in the STUDENT or INSTRUCTOR tables. Write two different solutions, using set operators for both.

2) Write a SQL statement, using a set operator, to show which students enrolled in a section on the same day they registered.

3) Find the students who are not enrolled in any classes. Write three solutions: a set operation, a subquery, and a correlated subquery.

4) Show the students who have received grades for their classes. Write four solutions: a set operation, a subquery, a correlated subquery, and a join.