Creating Efficient SQL

Structured Query Language is like any other programming language in that it can be coded well, coded poorly, and everywhere in between. Learning to create efficient SQL statements has been discussed in countless books. This chapter zeroes in on basic SQL coding fundamentals, and addresses some techniques to improve performance of your SQL statements. In addition, some emphasis is given to ramifications of poorly written SQL, along with a few common pitfalls to avoid in your SQL statements within your application.

Writing good SQL statements the first time is the best way to get good performance from your SQL queries. Knowing the fundamentals is the key to accomplishing the goal of good performance. This chapter focuses on the basics of the SQL language:

- SELECT statement
- WHERE clause
- Joining tables
- Subqueries
- Set operators

Then, we’ll focus on basic techniques to improve performance of your queries, as well as help ensure your queries are not hindering the performance of other queries within your database. It’s important to take the time to write efficient SQL statements the first time, which is easy to say, but tough to accomplish when balancing client requirements, budgets, and project timelines. However, if you adhere to basic coding practices and fundamentals, you can greatly improve the performance of your SQL queries.

**Note**  Several times in this chapter, we make a distinction between ISO syntax and traditional Oracle syntax. Specifically, we do that with respect to join syntax. However, that distinction is a bit mis-stated. With the exception of Oracle’s use of the (+) to indicate an outer join, all of Oracle’s join syntax complies with the ISO SQL standard, so it is all ISO syntax. However, it is common in the field to refer to the more newly implemented syntax as “ISO syntax,” and we follow that pattern in this chapter.
8-1. Retrieving All Rows from a Table

Problem

You need to write a query to retrieve all rows from a given table within your database.

Solution

Within the SQL language, you use the SELECT statement to retrieve data from the database. Everything following the SELECT statement tells Oracle what data you need from the database. The first thing you need to determine is from which table(s) you need to retrieve data. Once this has been determined, you have what you need to be able to run a query to get data from the database. If we have an EMPLOYEES table within our Oracle database, we can perform a describe on that table in order to see the structure of the table. By doing this, we can see the column names for the table, and can determine which columns we want to select from the database.

```
SQL> describe employees
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
EMPLOYEE_ID                               NOT NULL NUMBER(6)
FIRST_NAME                                         VARCHAR2(20)
LAST_NAME                                 NOT NULL VARCHAR2(25)
EMAIL                                     NOT NULL VARCHAR2(25)
PHONE_NUMBER                                       VARCHAR2(20)
HIRE_DATE                                 NOT NULL DATE
JOB_ID                                    NOT NULL VARCHAR2(10)
SALARY                                             NUMBER(8,2)
COMMISSION_PCT                                     NUMBER(2,2)
MANAGER_ID                                         NUMBER(6)
DEPARTMENT_ID                                      NUMBER(4)
```

If we want to retrieve a list of all the employees’ names from our EMPLOYEES table, we now have all the information we need to assemble a simple query against the EMPLOYEES table in the database. We know we are selecting from the EMPLOYEES table, which is needed for the FROM clause. We also know we want to select the names of the employees, which is needed to satisfy the SELECT clause. At this point, we can issue the following query against the database:

```
SELECT last_name, first_name
FROM employees;
```

<table>
<thead>
<tr>
<th>LAST_NAME</th>
<th>FIRST_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abel</td>
<td>Ellen</td>
</tr>
<tr>
<td>Baer</td>
<td>Hermann</td>
</tr>
<tr>
<td>Cabrio</td>
<td>Anthony</td>
</tr>
<tr>
<td>Dilly</td>
<td>Jennifer</td>
</tr>
<tr>
<td>Ernst</td>
<td>Bruce</td>
</tr>
</tbody>
</table>