Practice Exercises

- Online analytical processing (OLAP)
- Multidimensional data
  - Measure attributes
  - Dimension attributes
- Pivoting
- Data cube
- Slicing and dicing
- Rollup and drill down
- Cross-tabulation

Practice Exercises

5.1 Describe the circumstances in which you would choose to use embedded SQL rather than SQL alone or only a general-purpose programming language.

5.2 Write a Java function using JDBC metadata features that takes a ResultSet as an input parameter, and prints out the result in tabular form, with appropriate names as column headings.

5.3 Write a Java function using JDBC metadata features that prints a list of all relations in the database, displaying for each relation the names and types of its attributes.

5.4 Show how to enforce the constraint “an instructor cannot teach in two different classrooms in a semester in the same time slot.” using a trigger (remember that the constraint can be violated by changes to the teaches relation as well as to the section relation).

5.5 Write triggers to enforce the referential integrity constraint from section to timeslot, on updates to section, and timeslot. Note that the ones we wrote in Figure 5.8 do not cover the update operation.

5.6 To maintain the totcred attribute of the student relation, carry out the following:
   a. Modify the trigger on updates of takes, to handle all updates that can affect the value of totcred.
   b. Write a trigger to handle inserts to the takes relation.
   c. Under what assumptions is it reasonable not to create triggers on the course relation?

5.7 Consider the bank database of Figure 5.25. Let us define a view branch_cust as follows:

```sql
create view branch_cust as
select branch_name, customer_name
from depositor, account
where depositor.account_number = account.account_number
```
Suppose that the view is materialized; that is, the view is computed and stored. Write triggers to maintain the view, that is, to keep it up-to-date on insertions to and deletions from depositor or account. Do not bother about updates.

5.8 Consider the bank database of Figure 5.25. Write an SQL trigger to carry out the following action: On delete of an account, for each owner of the account, check if the owner has any remaining accounts, and if she does not, delete her from the depositor relation.

5.9 Show how to express group by cube \((a, b, c, d)\) using rollup; your answer should have only one group by clause.

5.10 Given a relation \(S(stu\text{dent}, subject, marks)\), write a query to find the top \(n\) students by total marks, by using ranking.

5.11 Consider the sales relation from Section 5.6. Write an SQL query to compute the cube operation on the relation, giving the relation in Figure 5.21. Do not use the cube construct.

Exercises

5.12 Consider the following relations for a company database:

- \(emp(ename, dname, salary)\)
- \(mgr(ename, mname)\)

and the Java code in Figure 5.26, which uses the JDBC API. Assume that the userid, password, machine name, etc. are all okay. Describe in concise English what the Java program does. (That is, produce an English sentence like “It finds the manager of the toy department,” not a line-by-line description of what each Java statement does.)

5.13 Suppose you were asked to define a class MetaDisplay in Java, containing a method static void printTable(String \(r\)): the method takes a relation name \(r\) as input, executes the query “select * from \(r\)”, and prints the result out in nice tabular format, with the attribute names displayed in the header of the table.
import java.sql.*;
public class Mystery {
    public static void main(String[] args) {
        try {
            Connection con=null;
            Class.forName("oracle.jdbc.driver.OracleDriver");
            con=DriverManager.getConnection("jdbc:oracle:thin:star/X@//edgar.cse.lehigh.edu:1521/XE");
            Statement s=con.createStatement();
            String q;
            String empName = "dog";
            boolean more;
            ResultSet result;
            do {
                q = "select mname from mgr where ename = "+ empName + ";"
                result = s.executeQuery(q);
                more = result.next();
                if (more) {
                    empName = result.getString("mname");
                    System.out.println(empName);
                }
            } while (more);
            s.close();
            con.close();
        } catch(Exception e) {e.printStackTrace();}
    }

    Figure 5.26 Java code for Exercise 5.12.

    a. What do you need to know about relation r to be able to print the result in the specified tabular format.

    b. What JDBC methods(s) can get you the required information?

    c. Write the method printTable(String r) using the JDBC API.

5.14 Repeat Exercise 5.13 using ODBC, defining void printTable(char *r) as a function instead of a method.

5.15 Consider an employee database with two relations

\[
\begin{align*}
\text{employee} & (\text{employee\_name}, \text{street}, \text{city}) \\
\text{works} & (\text{employee\_name}, \text{company\_name}, \text{salary})
\end{align*}
\]

where the primary keys are underlined. Write a query to find companies whose employees earn a higher salary, on average, than the average salary at “First Bank Corporation”.
Chapter 5  Advanced SQL

a. Using SQL functions as appropriate.
b. Without using SQL functions.

5.16  Rewrite the query in Section 5.2.1 that returns the name and budget of all departments with more than 12 instructors, using the with clause instead of using a function call.

5.17  Compare the use of embedded SQL with the use in SQL of functions defined in a general-purpose programming language. Under what circumstances would you use each of these features?

5.18  Modify the recursive query in Figure 5.15 to define a relation

\[
\text{prereq-depth}(\text{course}_\text{id}, \text{prereq}_\text{id}, \text{depth})
\]

where the attribute \textit{depth} indicates how many levels of intermediate prerequisites are there between the course and the prerequisite. Direct prerequisites have a depth of 0.

5.19  Consider the relational schema

\[
\text{part} (\text{part}_\text{id}, \text{name}, \text{cost})
\]
\[
\text{subpart} (\text{part}_\text{id}, \text{subpart}_\text{id}, \text{count})
\]

A tuple \((p_1, p_2, 3)\) in the \textit{subpart} relation denotes that the part with part-id \(p_2\) is a direct subpart of the part with part-id \(p_1\), and \(p_1\) has 3 copies of \(p_2\). Note that \(p_2\) may itself have further subparts. Write a recursive SQL query that outputs the names of all subparts of the part with part-id “P-100”.

5.20  Consider again the relational schema from Exercise 5.19. Write a JDBC function using non-recursive SQL to find the total cost of part “P-100”, including the costs of all its subparts. Be sure to take into account the fact that a part may have multiple occurrences of a subpart. You may use recursion in Java if you wish.

5.21  Suppose there are two relations \(r\) and \(s\), such that the foreign key \(B\) of \(r\) references the primary key \(A\) of \(s\). Describe how the trigger mechanism can be used to implement the \textbf{on delete cascade} option, when a tuple is deleted from \(s\).

5.22  The execution of a trigger can cause another action to be triggered. Most database systems place a limit on how deep the nesting can be. Explain why they might place such a limit.

5.23  Consider the relation, \(r\), shown in Figure 5.27. Give the result of the following query:
5.24 For each of the SQL aggregate functions `sum`, `count`, `min`, and `max`, show how to compute the aggregate value on a multiset $S_1 \cup S_2$, given the aggregate values on multisets $S_1$ and $S_2$.

On the basis of the above, give expressions to compute aggregate values with grouping on a subset $S$ of the attributes of a relation $r(A, B, C, D, E)$, given aggregate values for grouping on attributes $T \supseteq S$, for the following aggregate functions:

a. `sum`, `count`, `min`, and `max`

b. `avg`

c. Standard deviation

5.25 In Section 5.5.1, we used the student grades view of Exercise 4.5 to write a query to find the rank of each student based on grade-point average. Modify that query to show only the top 10 students (that is, those students whose rank is 1 through 10).

5.26 Give an example of a pair of groupings that cannot be expressed by using a single `group by` clause with `cube` and `rollup`.

5.27 Given relation $s(a, b, c)$, show how to use the extended SQL features to generate a histogram of $c$ versus $a$, dividing $a$ into 20 equal-sized partitions (that is, where each partition contains 5 percent of the tuples in $s$, sorted by $a$).

5.28 Consider the bank database of Figure 5.25 and the `balance` attribute of the `account` relation. Write an SQL query to compute a histogram of `balance` values, dividing the range 0 to the maximum account balance present, into three equal ranges.