Database Systems: Design, Implementation, and Management

Lesson 8

Objectives

- In this chapter, you will learn:
  - How to use the advanced SQL JOIN operator syntax
  - About the different types of subqueries and correlated queries
  - How to use SQL functions to manipulate dates, strings, and other data
  - About the relational set operators UNION, UNION ALL, INTERSECT, and MINUS

Objectives (cont'd.)

- How to create and use views and updatable views
- How to create and use triggers and stored procedures

SQL Join Operators

- Join operation merges rows from two tables and returns the rows with one of the following:
  - Have common values in common columns
    - Natural join
  - Meet a given join condition
    - Equality or inequality
  - Have common values in common columns or have no matching values
    - Outer join
- Inner join: only returns rows meeting criteria

Cross Join

- Performs relational product of two tables
  - Also called Cartesian product
- Syntax:

  SELECT column-list FROM table1 CROSS JOIN table2
- Perform a cross join that yields specified attributes

Natural Join

- Returns all rows with matching values in the matching columns
  - Eliminates duplicate columns
- Used when tables share one or more common attributes with common names
- Syntax:

  SELECT column-list FROM table1 NATURAL JOIN table2

JOIN USING Clause

- Returns only rows with matching values in the column indicated in the USING clause
- Syntax:

  SELECT column-list FROM table1 JOIN table2 USING (common-column)
- JOIN USING operand does not require table qualifiers
  - Oracle returns error if table name is specified

JOIN ON Clause

- Used when tables have no common attributes
- Returns only rows that meet the join condition
  - Typically includes equality comparison expression of two columns
- Syntax:

  SELECT column-list FROM table1 JOIN table2 ON join-condition

Outer Joins

- Returns rows matching the join condition
- Also returns rows with unmatched attribute values for tables to be joined
- Three types
  - Left
  - Right
  - Full
- Left and right designate order in which tables are processed

Outer Joins (cont'd.)

- Left outer join
  - Returns rows matching the join condition
  - Returns rows in left side table with unmatched values
  - Syntax: SELECT column-list FROM table1 LEFT [OUTER] JOIN table2 ON join-condition
- Right outer join
  - Returns rows matching join condition
  - Returns rows in right side table with unmatched values

Outer Joins (cont'd.)

- Full outer join
  - Returns rows matching join condition
  - Returns all rows with unmatched values in either side table
  - Syntax:

```
SELECT column-list
FROM   table1 FULL [OUTER] JOIN table2
           ON join-condition
```

Subqueries and Correlated Queries

- Often necessary to process data based on other processed data
- Subquery is a query inside a query, normally inside parentheses
- First query is the outer query
  - Inside query is the inner query
- Inner query is executed first
- Output of inner query is used as input for outer query
- Sometimes referred to as a nested query

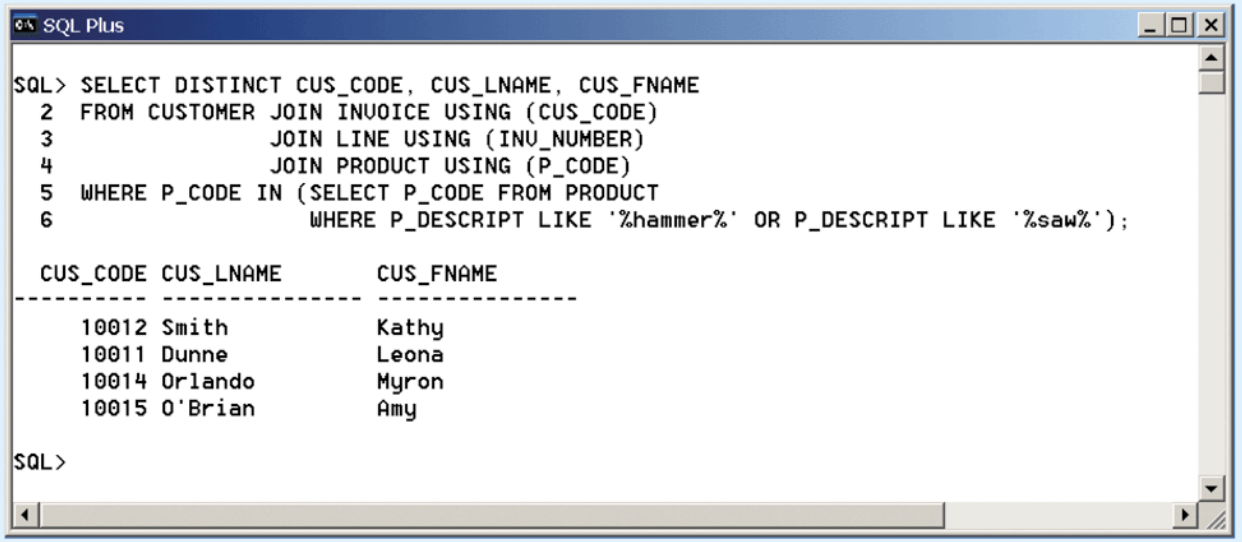
WHERE Subqueries

- Most common type uses inner SELECT subquery on right side of WHERE comparison
  - Requires a subquery that returns only one single value
- Value generated by subquery must be of comparable data type
- Can be used in combination with joins

IN Subqueries

- Used when comparing a single attribute to a list of values

FIGURE 8.8 IN subquery example

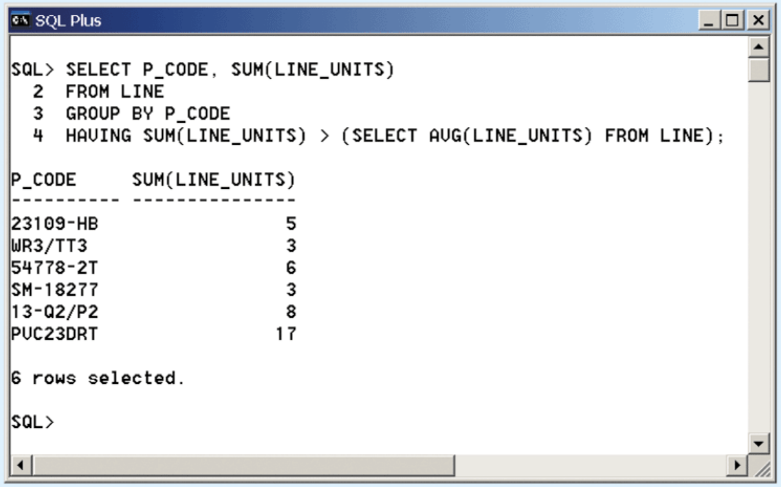


SOURCE: Course Technology/Cengage Learning

HAVING Subqueries

- HAVING clause restricts the output of a GROUP BY query
  - Applies conditional criterion to the grouped rows

FIGURE 8.9 HAVING subquery example



SOURCE: Course Technology/Cengage Learning

Multirow Subquery Operators: ANY and ALL

- Allows comparison of single value with a list of values using inequality comparison
- “Greater than ALL” equivalent to “greater than the highest in list”
- “Less than ALL” equivalent to “less than lowest”
- Using equal to ANY operator equivalent to IN operator

Attribute List Subqueries

- SELECT statement uses attribute list to indicate columns to project resulting set
  - Columns can be attributes of base tables
  - Result of aggregate function
- Attribute list can also include subquery expression: inline subquery
  - Must return one single value
- Cannot use an alias in the attribute list

SQL Functions

- Generating information from data often requires many data manipulations
- SQL functions are similar to functions in programming languages
- Functions always use numerical, date, or string value
- Value may be part of a command or attribute in a table
- Function may appear anywhere in an SQL statement

Date and Time Functions

- All SQL-standard DBMSs support date and time functions
- Date functions take one parameter and return a value
- Date/time data types are implemented differently by different DBMS vendors
- ANSI SQL standard defines date data types, but not how data types are stored

Numeric Functions

- Grouped in different ways
  - Algebraic, trigonometric, logarithmic, etc.
- Do not confuse with aggregate functions
  - Aggregate functions operate over sets
  - Numeric functions operate over single row
- Numeric functions take one numeric parameter and return one value

TABLE 8.5 **Selected Numeric Functions**

FUNCTION	EXAMPLE(S)
ABS Returns the absolute value of a number Syntax: ABS(numeric_value)	In Oracle, use the following: SELECT 1.95, -1.93, ABS(1.95), ABS(-1.93) FROM DUAL; In MS Access and SQL Server, use the following: SELECT 1.95, -1.93, ABS(1.95), ABS(-1.93);
ROUND Rounds a value to a specified precision (number of digits) Syntax: ROUND(numeric_value, p) p = precision	Lists the product prices rounded to one and zero decimal places: SELECT P_CODE, P_PRICE, ROUND(P_PRICE,1) AS PRICE1, ROUND(P_PRICE,0) AS PRICE0 FROM PRODUCT;
CEIL/CEILING/FLOOR Returns the smallest integer greater than or equal to a number or returns the largest integer equal to or less than a number, respectively Syntax: CEIL(numeric_value) Oracle CEILING(numeric_value) SQL Server FLOOR(numeric_value)	Lists the product price, the smallest integer greater than or equal to the product price, and the largest integer equal to or less than the product price. In Oracle, use the following: SELECT P_PRICE, CEIL(P_PRICE), FLOOR(P_PRICE) FROM PRODUCT; In SQL Server, use the following: SELECT P_PRICE, CEILING(P_PRICE), FLOOR(P_PRICE) FROM PRODUCT; MS Access does not support these functions.

String Functions

- String manipulations are the most used functions in programming
- String manipulation function examples:
  - Concatenation
  - Printing in uppercase
  - Finding length of an attribute

Relational Set Operators

- UNION
- INTERSECT
- MINUS
- Work properly if relations are union-compatible
  - Names of relation attributes must be the same and their data types must be identical

UNION

- Combines rows from two or more queries without including duplicate rows
  - Example:

```
SELECT   CUS_LNAME, CUS_FNAME,
   CUS_INITIAL, CUS_AREACODE,
FROM              CUSTOMER
UNION
SELECT   CUS_LNAME, CUS_FNAME,
   CUS_INITIAL, CUS_AREACODE,
FROM              CUSTOMER_2
```

- Can be used to unite more than two queries

UNION ALL

- Produces a relation that retains duplicate rows
  - Example query:

```
SELECT  CUS_LNAME, CUS_FNAME,
        CUS_INITIAL, CUS_AREACODE, FROM
   CUSTOMER
UNION ALL
SELECT  CUS_LNAME, CUS_FNAME,
        CUS_INITIAL, CUS_AREACODE, FROM
   CUSTOMER_2;
```

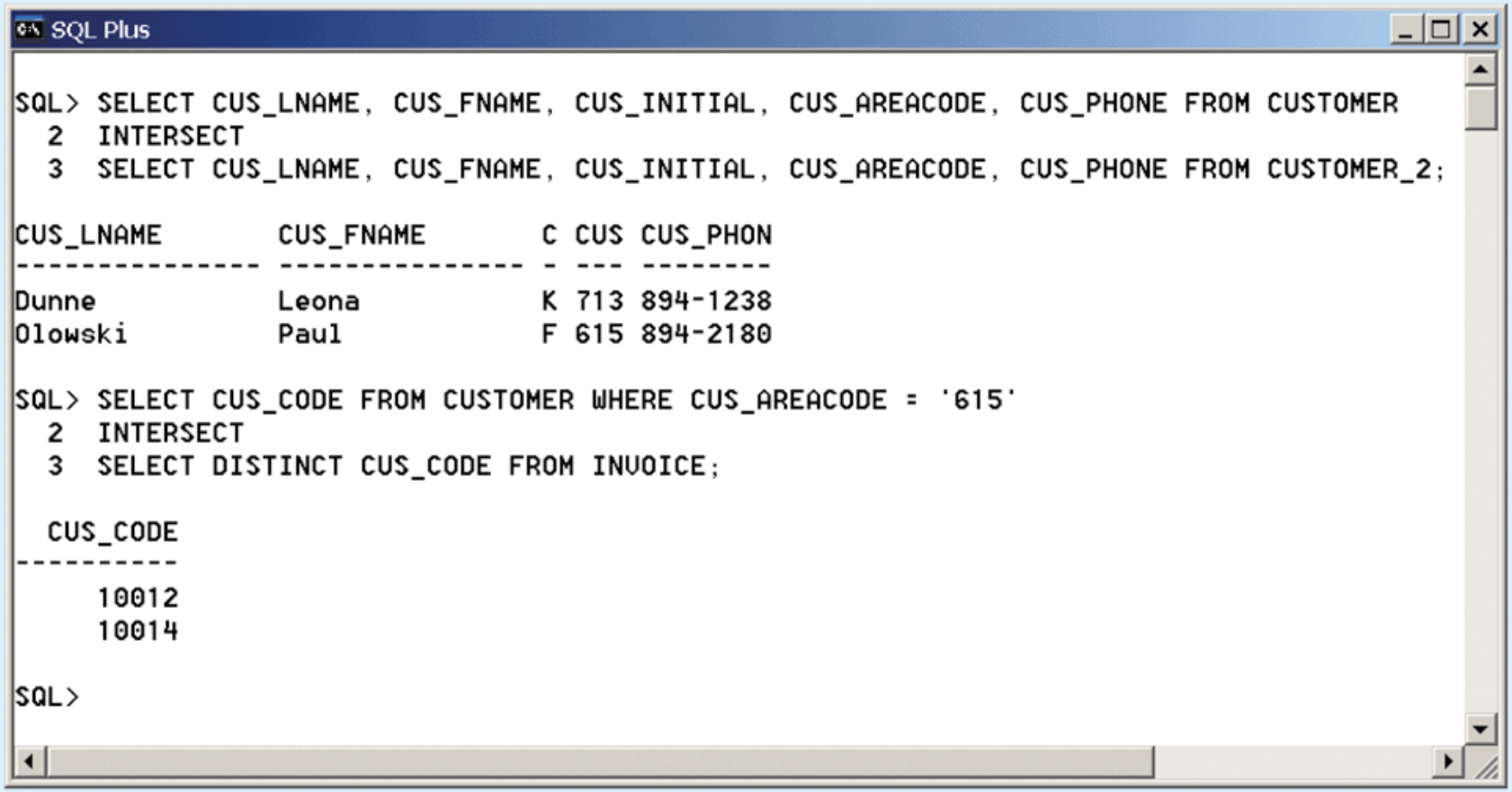
- Can be used to unite more than two queries

INTERSECT

- Combines rows from two queries, returning only the rows that appear in both sets
- Syntax: query INTERSECT query
  - Example query:

```
SELECT   CUS_LNAME, CUS_FNAME,
         CUS_INITIAL, CUS_AREACODE, FROM
   CUSTOMER
INTERSECT
SELECT   CUS_LNAME, CUS_FNAME,
         CUS_INITIAL, CUS_AREACODE, FROM
   CUSTOMER_2
```

FIGURE 8.18 INTERSECT query results



SOURCE: Course Technology/Cengage Learning

MINUS

- Combines rows from two queries
  - Returns only the rows that appear in the first set but not in the second
- Syntax: query MINUS query
  - Example:

```
SELECT CUS_LNAME, CUS_FNAME,
            CUS_INITIAL, CUS_AREACODE, FROM
  CUSTOMER
MINUS
SELECT  CUS_LNAME, CUS_FNAME,
            CUS_INITIAL, CUS_AREACODE, FROM
  CUSTOMER_2
```

Syntax Alternatives

- IN and NOT IN subqueries can be used in place of INTERSECT
- Example:

```
SELECT          CUS_CODE FROM CUSTOMER
WHERE           CUS_AREACODE = '615' AND
                CUS_CODE IN (SELECT
          DISTINCT CUS_CODE
                FROM
                INVOICE);
```

Virtual Tables: Creating a View

- View
  - Virtual table based on a SELECT query
- Base tables
- Tables on which the view is based
- CREATE VIEW viewname AS SELECT query
- Relational view special characteristics

Updatable Views

- Batch update routine pools multiple transactions into a single batch
  - Update master table field in a single operation
- Updatable view is a view that can be used to update attributes in the base tables
- Not all views are updatable
  - GROUP BY expressions or aggregate functions cannot be used
  - Cannot use set operators
  - Most restrictions are based on use of JOINs

Triggers

- Procedural SQL code automatically invoked by RDBMS on data manipulation event
- Trigger definition:
  - Triggering timing: BEFORE or AFTER
  - Triggering event: INSERT, UPDATE, DELETE
  - Triggering level:
    - Statement-level trigger
    - Row-level trigger
  - Triggering action
- DROP TRIGGER trigger_name

Stored Procedures

- Named collection of procedural and SQL statements
- Advantages
  - Substantially reduce network traffic and increase performance
    - No transmission of individual SQL statements over network
  - Reduce code duplication by means of code isolation and code sharing
    - Minimize chance of errors and cost of application development and maintenance