Objectives

• Determine when it is appropriate to use a subquery
• Identify which clauses can contain subqueries
• Distinguish between an outer query and a subquery
• Use a single-row subquery in a WHERE clause
• Use a single-row subquery in a HAVING clause
• Use a single-row subquery in a SELECT clause
Objectives (continued)

• Distinguish between single-row and multiple-row comparison operators
• Use a multiple-row subquery in a WHERE clause
• Use a multiple-row subquery in a HAVING clause
• Use a multiple-column subquery in a WHERE clause
Objectives (continued)

- Create an inline view using a multiple-column subquery in a FROM clause
- Compensate for NULL values in subqueries
- Distinguish between correlated and uncorrelated subqueries
- Nest a subquery inside another subquery
- Process multiple DML actions with a MERGE statement
Subqueries and Their Uses

• Subquery - a query nested inside another query
• Used when a query is based on an unknown value
• Requires SELECT and FROM clauses
• Must be enclosed in parentheses
• Place on right side of comparison operator
# Types of Subqueries

<table>
<thead>
<tr>
<th>Subquery</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-row subquery</td>
<td>Returns to the outer query one row of results that consists of one column</td>
</tr>
<tr>
<td>Multiple-row subquery</td>
<td>Returns to the outer query more than one row of results</td>
</tr>
<tr>
<td>Multiple-column subquery</td>
<td>Returns to the outer query more than one column of results</td>
</tr>
<tr>
<td>Correlated subquery</td>
<td>References a column in the outer query, and executes the subquery once for every row in the outer query</td>
</tr>
<tr>
<td>Uncorrelated subquery</td>
<td>Executes the subquery first and passes the value to the outer query</td>
</tr>
</tbody>
</table>
Single-Row Subqueries

- Can only return one result to the outer query
- Operators include =, >, <, >=, <=, <>
Single-Row Subquery in a WHERE Clause

- Used for comparison against individual data

```
SELECT title, cost
FROM books
WHERE cost > (SELECT cost
              FROM books
              WHERE title = 'DATABASE IMPLEMENTATION')
AND category = 'COMPUTER'
```

**FIGURE 12-4** A single-row subquery
Single-Row Subquery in a HAVING Clause

- Required when returned value is compared to grouped data

```
SELECT category, AVG(retail-cost) "Average Profit"
FROM books
GROUP BY category
HAVING AVG(retail-cost) >
(SELECT AVG(retail-cost)
 FROM books
 WHERE category = 'LITERATURE');
```

**FIGURE 12-8** Single-row subquery nested in a HAVING clause
Single-Row Subquery in a SELECT Clause

- Replicates subquery value for each row displayed

FIGURE 12-9  Single-row subquery in a SELECT clause
Multiple-Row Subqueries

• Return more than one row of results
• Require use of IN, ANY, ALL, or EXISTS operators
ANY and ALL Operators

- Combine with arithmetic operators

<table>
<thead>
<tr>
<th>OPERATOR</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;ALL</td>
<td>More than the highest value returned by the subquery</td>
</tr>
<tr>
<td>&lt;ALL</td>
<td>Less than the lowest value returned by the subquery</td>
</tr>
<tr>
<td>&lt;ANY</td>
<td>Less than the highest value returned by the subquery</td>
</tr>
<tr>
<td>&gt;ANY</td>
<td>More than the lowest value returned by the subquery</td>
</tr>
<tr>
<td>=ANY</td>
<td>Equal to any value returned by the subquery (same as IN)</td>
</tr>
</tbody>
</table>

FIGURE 12-11 Descriptions of ALL and ANY operator combinations
Multiple-Row Subquery in a WHERE Clause

Note: Could use IN operator or =ANY
Multiple-Row Subquery in a WHERE Clause (continued)

```sql
SELECT title, retail
FROM books
WHERE retail < ANY
(SELECT retail
FROM books
WHERE category = 'COOKING');
```

**Figure 12-15**  The < ANY operator
EXISTS Operator

- Determines whether condition exists in subquery

**FIGURE 12-18** Subquery using the EXISTS operator
Multiple-Row Subquery in a HAVING Clause

```
SELECT order#, SUM(retail*quantity)
FROM orders JOIN orderitems USING(order#)
    JOIN books USING(isbn)
HAVING SUM(retail*quantity) >
    ANY
    (SELECT AVG(SUM(quantity*retail))
     FROM orders JOIN orderitems USING(order#)
                JOIN books USING(isbn)
                GROUP BY shipstate)
GROUP BY order#;
```

**FIGURE 12-20** Multiple-row subquery in a HAVING clause
Multiple-Column Subqueries

• Return more than one column in results
• Can return more than one row
• Column list on the left side of operator must be in parentheses
• Use the IN operator for WHERE and HAVING clauses
Multiple-Column Subquery in a FROM Clause

• Creates a temporary table

FIGURE 12-23 Using a join with a multiple-column subquery in the FROM clause
Multiple-Column Subquery in a WHERE Clause

- Returns multiple columns for evaluation

**FIGURE 12-24**  Multiple-column subquery in a WHERE clause
NULL Values

- When a subquery might return NULL values, use NVL function
Uncorrelated Subqueries

• Processing sequence:
  – Inner query is executed first
  – Result is passed to outer query
  – Outer query is executed
Correlated Subqueries

- Inner query is executed once for each row processed by the outer query
- Inner query references the row contained in the outer query
Correlated Subqueries (continued)

```sql
SELECT title
FROM books
WHERE EXISTS
    (SELECT isbn
     FROM orderitems
     WHERE books.isbn = orderitems.isbn);
```

![Correlated subquery example](image)

**FIGURE 12-28** Correlated subquery
Nested Subqueries

- Maximum of 255 subqueries if nested in the WHERE clause
- No limit if nested in the FROM clause
- Innermost subquery is resolved first, then the next level, etc.
Nested Subqueries

- Innermost is resolved first (3), then the second level (2), then the outer query (1)
MERGE Statement

- With a MERGE statement, a series of DML actions can occur with a single SQL statement
- Conditionally updates one data source based on another
MERGE Statement (continued)

Conditions comparing the two data sources

FIGURE 12-32  MERGE statement with UPDATE and INSERT
Oracle 10g: SQL
MERGE Statement (continued)

• The following explains each part of the previous MERGE statement:
  • MERGE INTO books_1 a: The BOOKS_1 table is to be changed and a table alias of “a” is assigned to this table
  • USING books_2 b: The BOOKS_2 table will provide the data to update and/or insert into BOOKS_1 and a table alias of “b” is assigned to this table
  • ON (a.isbn = b.isbn): The rows of the two tables will be joined or matched based on isbn
  • WHEN MATCHED THEN: If a row match based on ISBN is discovered, execute the UPDATE action in this clause. The UPDATE action instructs the system to modify only two columns (Retail and Category)
  • WHEN NOT MATCHED THEN: If no match is found based on the ISBN (a books exists in BOOKS_2 that is not in BOOKS_1), then perform the INSERT action in this clause
MERGE with WHERE conditions

```
MERGE INTO books_1 a
USING books_2 b
ON (a.isbn = b.isbn)
WHEN MATCHED THEN
  UPDATE SET a.retail = b.retail, a.category = b.category
  WHERE b.category = 'COMPUTER'
WHEN NOT MATCHED THEN
  INSERT (isbn, title, pubdate, retail, category)
  VALUES (b.isbn, b.title, b.pubdate, b.retail, b.category)
  WHERE b.category = 'COMPUTER';
```

FIGURE 12-33 Using WHERE conditions in a MERGE statement
MERGE with DELETE

```
MERGE INTO books_1 a
USING books_2 b
ON (a.isbn = b.isbn)
WHEN MATCHED THEN
  UPDATE SET a.retail = b.retail, a.category = b.category
DELETE WHERE (b.retail < 50);

SELECT *
FROM books_1;
```

3 rows merged.

**FIGURE 12-34** Conditional DELETE in a MERGE statement
Summary

• A subquery is a complete query nested in the SELECT, FROM, HAVING, or WHERE clause of another query
  – The subquery must be enclosed in parentheses and have a SELECT and a FROM clause, at a minimum
• Subqueries are completed first; the result of the subquery is used as input for the outer query
• A single-row subquery can return a maximum of one value
• Single-row operators include =, >, <, >=, <=, and <>
• Multiple-row subqueries return more than one row of results
Summary (continued)

- Operators that can be used with multiple-row subqueries include IN, ALL, ANY, and EXISTS
- Multiple-column subqueries return more than one column to the outer query
- NULL values returned by a multiple-row or multiple-column subquery will not present a problem if the IN or =ANY operator is used
- Correlated subqueries reference a column contained in the outer query
- Subqueries can be nested to a maximum depth of 255 subqueries in the WHERE clause of the parent query
Summary (continued)

• With nested subqueries, the innermost subquery is executed first, then the next highest level subquery is executed, and so on, until the outermost query is reached.

• A MERGE statement allows multiple DML actions to be conditionally performed while comparing data of two tables.