SELF TEST ANSWERS

Write SELECT Statements to Access Data from More Than One Table Using Equijoins and Nonequijoins

1. ☑ **D.** The queries in **B** and **C** incorrectly contain the NATURAL keyword. If this is removed, they will join the DEPARTMENTS and EMPLOYEES tables based on the DEPARTMENT_ID column.
   ☒ **A, B,** and **C** are incorrect. **A** performs a pure natural join that implicitly joins the two tables on all columns with identical names which, in this case, are DEPARTMENT_ID and MANAGER_ID.
2. ☑ **A, B,** and **C.** These clauses demonstrate different techniques to join the tables on both the DEPARTMENT_ID and MANAGER_ID columns.
   ☒ **D** is incorrect.
3. ☑ **B.** The join condition is an expression based on the *less than* inequality operator. Therefore, this join is a nonequijoin.
   ☒ **A, C,** and **D** are incorrect. **A** would be correct if the operator in the join condition expression was an equality operator. The CROSS JOIN keywords or the absence of a join condition would result in **C** being true. **D** would be true if one of the OUTER JOIN clause was used instead of the JOIN…ON clause.
4. ☑ **A.** This statement demonstrates the correct usage of the JOIN…USING clause.
   ☒ **B, C,** and **D** are incorrect. **B** is incorrect since only nonqualified column names are allowed in the brackets after the USING keyword. **C** is incorrect because the column in brackets after the USING keyword cannot be referenced with a qualifier in the SELECT clause.
5. ☑ **B** demonstrates the correct usage of the JOIN…ON clause.
   ☒ **A, C,** and **D** are incorrect. **A** is incorrect since the CROSS JOIN clause cannot contain the ON keyword. **C** is incorrect since the OUTER JOIN keywords must be preceded by the LEFT, RIGHT, or FULL keyword.
6. ☑ **C.** The JOIN…ON clause and the other join clauses may all be used for joins between multiple tables. The JOIN…ON and JOIN…USING clauses are better suited for N-way table joins.
   ☒ **A, B,** and **D** are incorrect. **A** is false since you may join as many tables as you wish. A Cartesian product is not created since there are two join conditions and three tables.

Join a Table to Itself Using a Self-Join

7. ☑ **B.** Three rows are returned. For the row with a REGION_ID value of 2, the REGION_NAME is Asia and half the length of the REGION_NAME is also 2. Therefore this row is returned.

The same logic results in the rows with REGION_ID values of three and four and REGION_NAME values of Europe and Americas being returned.
☒ **A, C,** and **D** are incorrect.

View Data That Does Not Meet a Join Condition Using Outer Joins

8. ☑ **A.** The right outer join fetches the COUNTRIES rows that the inner join between the LOCATIONS and COUNTRIES tables have excluded. The WHERE clause then restricts the results by eliminating the inner join results. This leaves the rows from the COUNTRIES table with which no records from the LOCATIONS table records are associated.
☒ **B, C,** and **D** are incorrect.

9. ☑ **A.** This statement demonstrates the correct use of the RIGHT OUTER JOIN…ON clause.
☒ **B, C,** and **D** are incorrect. The JOB_ID column in the SELECT clause in **B** is not qualified and is therefore ambiguous since the table from which this column comes is not specified. **C** uses an OUTER JOIN without the keywords LEFT, RIGHT, or FULL.

Generate a Cartesian Product of Two or More Tables

10. ☑ **A.** The cross join associates every four rows from the REGIONS table 25 times with the rows from the COUNTRIES table yielding a result set that contains 100 rows.
☒ **B, C,** and **D** are incorrect.

LAB ANSWER

Using SQL Developer or SQL*Plus, connect to the OE schema, and complete the following tasks. There are several approaches to solving this question. Your approach may differ from the following solution listed.

1. Start SQL Developer and connect to the OE schema.
2. The SELECT list consists of four columns from two tables, which will be associated with each other using several joins. The SELECT clause is
```
SELECT CUST_FIRST_NAME, CUST_LAST_NAME, PRODUCT_NAME, LIST_PRICE.
```
3. The FROM clause is
```
FROM CUSTOMERS.
```
4. The WHERE clause is
```
WHERE LIST_PRICE > 1000.
```
5. The JOIN clauses are interesting since the PRODUCT_INFORMATION and CUSTOMERS tables not directly related. They are related through two other tables.