Oracle_CertifyMe_1Z0-051_v2011-05-18_180q_By-steeve

Number: 1Z0-051
Passing Score: 600
Time Limit: 120 min
File Version: 2011-05-18

Oracle 1Z0-051

Questions: 180
Version: 2011-05-18
Good luck Friends!

By Steeve

Corrected by vlak

Exam A

QUESTION 1
View the Exhibit and examine the structure of the SALES, CUSTOMERS, PRODUCTS, and TIMES tables.

The PROD_ID column is the foreign key in the SALES table, which references the PRODUCTS table.

Similarly, the CUST_ID and TIME_ID columns are also foreign keys in the SALES table referencing the

CUSTOMERS and TIMES tables, respectively.

Evaluate the following CREATE TABLE command:

```
CREATE TABLE new_sales(prod_id, cust_id, order_date DEFAULT SYSDATE)
AS
SELECT prod_id, cust_id, time_id
FROM sales;
```

Which statement is true regarding the above command?

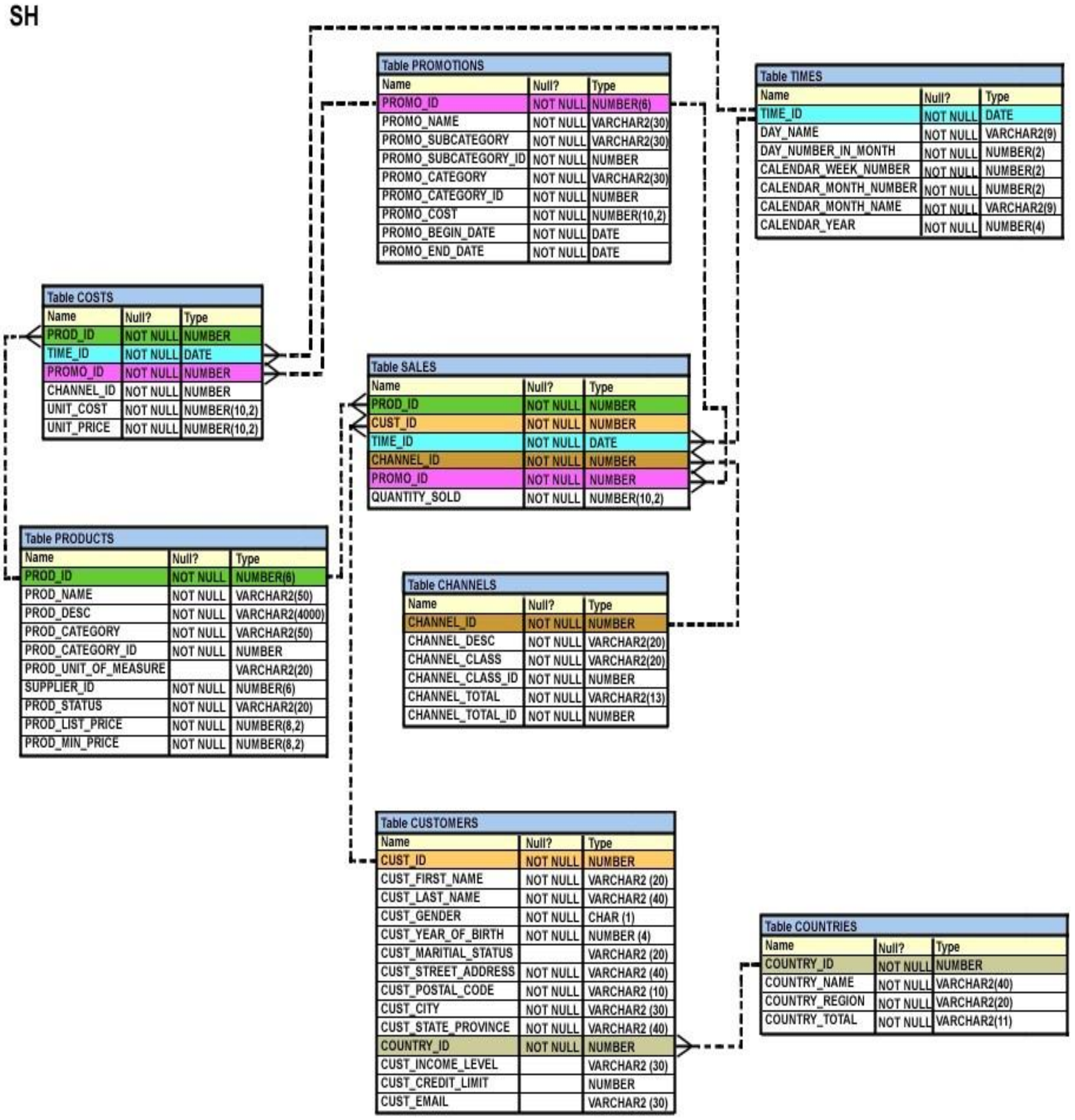


Table PROMOTIONS

Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

Table TIMES

Name	Null?	Type
TIME_ID	NOT NULL	DATE
DAY_NAME	NOT NULL	VARCHAR2(9)
DAY_NUMBER_IN_MONTH	NOT NULL	NUMBER(2)
CALENDAR_WEEK_NUMBER	NOT NULL	NUMBER(2)
CALENDAR_MONTH_NUMBER	NOT NULL	NUMBER(2)
CALENDAR_MONTH_NAME	NOT NULL	VARCHAR2(9)
CALENDAR_YEAR	NOT NULL	NUMBER(4)

Table COSTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
PROMO_ID	NOT NULL	NUMBER
CHANNEL_ID	NOT NULL	NUMBER
UNIT_COST	NOT NULL	NUMBER(10,2)
UNIT_PRICE	NOT NULL	NUMBER(10,2)

Table SALES

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
CUST_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
CHANNEL_ID	NOT NULL	NUMBER
PROMO_ID	NOT NULL	NUMBER
QUANTITY_SOLD	NOT NULL	NUMBER(10,2)

Table PRODUCTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

Table CHANNELS

Name	Null?	Type
CHANNEL_ID	NOT NULL	NUMBER
CHANNEL_DESC	NOT NULL	VARCHAR2(20)
CHANNEL_CLASS	NOT NULL	VARCHAR2(20)
CHANNEL_CLASS_ID	NOT NULL	NUMBER
CHANNEL_TOTAL	NOT NULL	VARCHAR2(13)
CHANNEL_TOTAL_ID	NOT NULL	NUMBER

Table CUSTOMERS

Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

Table COUNTRIES

Name	Null?	Type
COUNTRY_ID	NOT NULL	NUMBER
COUNTRY_NAME	NOT NULL	VARCHAR2(40)
COUNTRY_REGION	NOT NULL	VARCHAR2(20)
COUNTRY_TOTAL	NOT NULL	VARCHAR2(11)

A. The NEW_SALES table would not get created because the DEFAULT value cannot be specified in the column definition.
B. The NEW_SALES table would get created and all the NOT NULL constraints defined on the specified columns would be passed to the new table.
C. The NEW_SALES table would not get created because the column names in the CREATE TABLE command and the SELECT clause do not match.
D. The NEW_SALES table would get created and all the FOREIGN KEY constraints defined on the specified columns would be passed to the new table.

Answer: B
Section: (none)

Explanation/Reference:

Creating a Table Using a Subquery

Create a table and insert rows by combining the CREATE TABLE statement and the AS **subquery** option.

```
CREATE TABLE table
[(column, column...)]
AS subquery;
```

Match the number of specified columns to the number of subquery columns.

Define columns with column names and default values.

Guidelines

The table is created with the specified column names, and the rows retrieved by the SELECT statement are inserted into the table.

The column definition can contain only the column name and default value.

If column specifications are given, the number of columns must equal the number of columns in the subquery SELECT list.

If no column specifications are given, the column names of the table are the same as the column names in the subquery.

The column data type definitions and the NOT NULL constraint are passed to the new table. Note that only the explicit NOT NULL constraint will be inherited. The PRIMARY KEY column will not pass the NOT NULL feature to the new column. Any other constraint rules are not passed to the new table. However, you can add constraints in the column definition.

QUESTION 2

View the Exhibit to examine the description for the SALES table.

Which views can have all DML operations performed on it? (Choose all that apply.)

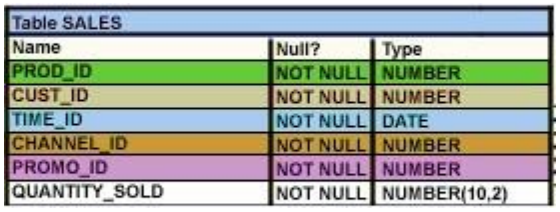


Table SALES		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
CUST_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
CHANNEL_ID	NOT NULL	NUMBER
PROMO_ID	NOT NULL	NUMBER
QUANTITY_SOLD	NOT NULL	NUMBER(10,2)

A. CREATE VIEW v3
AS SELECT * FROM SALES
WHERE cust_id = 2034
WITH CHECK OPTION;

B. CREATE VIEW v1
AS SELECT * FROM SALES
WHERE time_id <= SYSDATE - 2*365
WITH CHECK OPTION;

C. CREATE VIEW v2
AS SELECT prod_id, cust_id, time_id FROM SALES
WHERE time_id <= SYSDATE - 2*365
WITH CHECK OPTION;

D. CREATE VIEW v4
AS SELECT prod_id, cust_id, SUM(quantity_sold) FROM SALES
WHERE time_id <= SYSDATE - 2*365
GROUP BY prod_id, cust_id
WITH CHECK OPTION;

Answer: AB
Section: (none)

Explanation/Reference:

Creating a View

You can create a view by embedding a subquery in the CREATE VIEW statement.
In the syntax:

CREATE [OR REPLACE] [FORCE|NOFORCE] VIEW view
[(alias[, alias]...)]
AS subquery
[WITH CHECK OPTION [CONSTRAINT constraint]]
[WITH READ ONLY [CONSTRAINT constraint]];

OR REPLACE Re-creates the view if it already exists
FORCE Creates the view regardless of whether or not the base tables exist
NOFORCE Creates the view only if the base tables exist (This is the default.)
view Is the name of the view
alias Specifies names for the expressions selected by the view's query (The number of aliases must match the number of expressions selected by the view.)
subquery Is a complete SELECT statement (You can use aliases for the columns in the SELECT list.)
<u>WITH CHECK OPTION Specifies that only those rows that are accessible to the view can be inserted or updated ANSWER D</u>
constraint Is the name assigned to the CHECK OPTION constraint
WITH READ ONLY Ensures that no DML operations can be performed on this view

Rules for Performing DML Operations on a View

You cannot add data through a view if the view includes:

Group functions
A GROUP BY clause
The DISTINCT keyword
The pseudocolumn ROWNUM keyword
Columns defined by expressions
<u>NOT NULL columns in the base tables that are not selected by the view – ANSWER C</u>

QUESTION 3
You need to extract details of those products in the SALES table where the PROD_ID column contains the string '_D123'.

Which WHERE clause could be used in the SELECT statement to get the required output?

A. WHERE prod_id LIKE '%_D123%' ESCAPE '_'
B. WHERE prod_id LIKE '%_D123%' ESCAPE '\'
C. WHERE prod_id LIKE '%_D123%' ESCAPE '%_'
D. WHERE prod_id LIKE '%_D123%' ESCAPE '_'

Answer: B
Section: (none)

Explanation/Reference:
A naturally occurring underscore character may be escaped (or treated as a regular nonspecial symbol) using the ESCAPE identifier in conjunction with an ESCAPE character. The second example in Figure 3-12 shows the SQL statement that retrieves the JOBS table records with JOB_ID values equal to SA_MAN and SA_REP and which conforms to the original requirement:
select job_id from jobs
where job_id like 'SA_%' escape '\';

QUESTION 4
Which two statements are true regarding single row functions? (Choose two.)

A. They accept only a single argument.
B. They can be nested only to two levels.
C. Arguments can only be column values or constants.
D. They always return a single result row for every row of a queried table.
E. They can return a data type value different from the one that is referenced.

Answer: DE
Section: (none)

Explanation/Reference:
A function is a program written to optionally accept input parameters, perform an operation, or return a single value. A function returns only one value per execution.
Three important components form the basis of defining a function. The first is the input parameter list. It specifies zero or more arguments that may be passed to a function as input for processing. These arguments or parameters may be of differing data types, and some are mandatory while others may be optional. The second component is the data type of its resultant value. Upon execution, only one value is returned by the function. The third encapsulates the details of the processing performed by the function and contains the program code that optionally manipulates the input parameters, performs calculations and operations, and generates a return value.

QUESTION 5
Which SQL statements would display the value 1890.55 as $1,890.55? (Choose three .)

A. SELECT TO_CHAR(1890.55,'$0G000D00')
   FROM DUAL;
B. SELECT TO_CHAR(1890.55,'$9,999V99')
   FROM DUAL;
C. SELECT TO_CHAR(1890.55,'$99,999D99')
   FROM DUAL;
D. SELECT TO_CHAR(1890.55,'$99G999D00')
   FROM DUAL;
E. SELECT TO_CHAR(1890.55,'$99G999D99')
   FROM DUAL;

Answer: ADE
Section: (none)

Explanation/Reference:

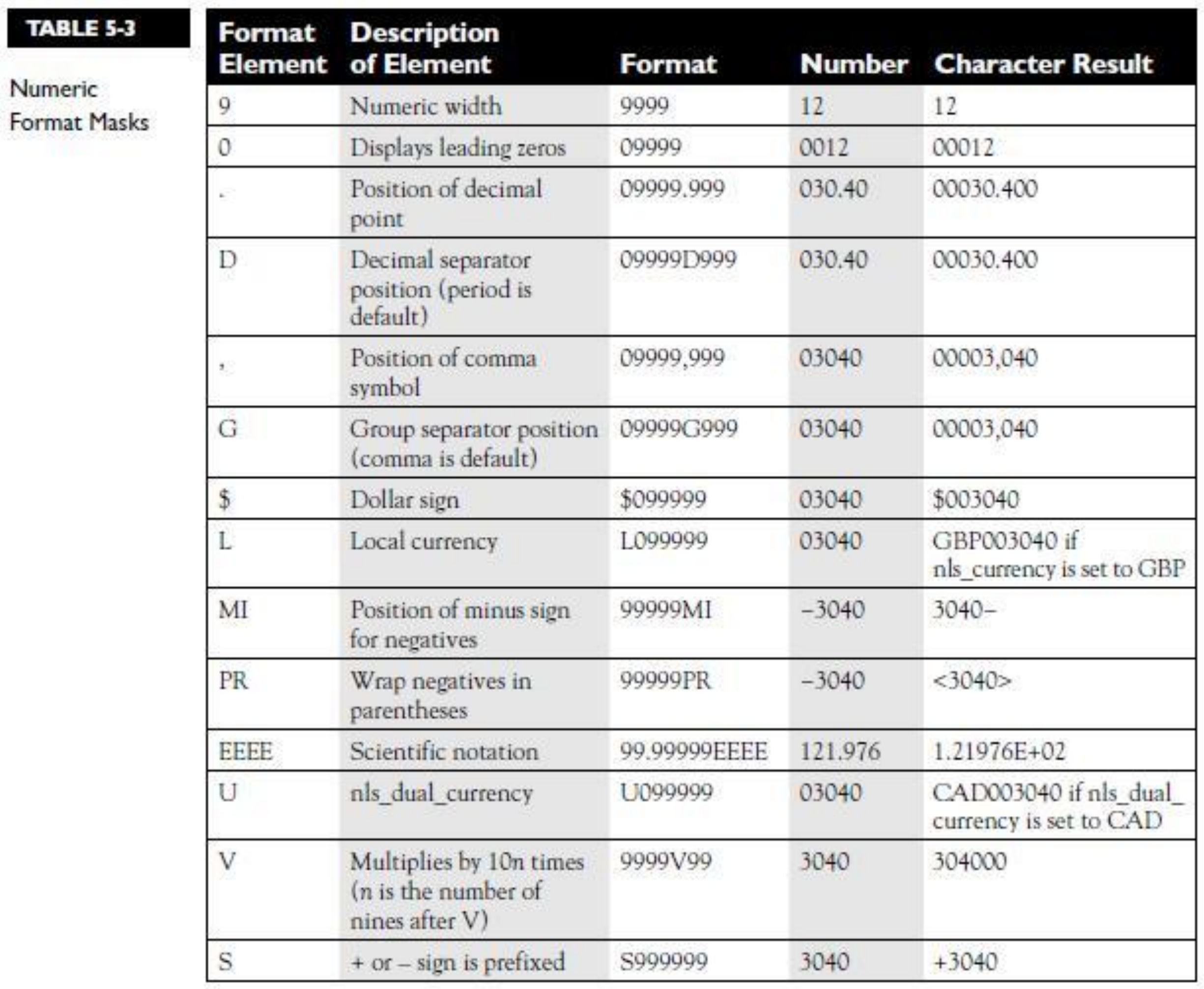


TABLE 5-3

Numeric Format Masks

Format Element	Description of Element	Format	Number	Character Result
9	Numeric width	9999	12	12
0	Displays leading zeros	09999	0012	00012
.	Position of decimal point	09999.999	030.40	00030.400
D	Decimal separator position (period is default)	09999D999	030.40	00030.400
,	Position of comma symbol	09999,999	03040	00003,040
G	Group separator position (comma is default)	09999G999	03040	00003,040
$	Dollar sign	$099999	03040	$003040
L	Local currency	L099999	03040	GBP003040 if nls_currency is set to GBP
MI	Position of minus sign for negatives	99999MI	–3040	3040–
PR	Wrap negatives in parentheses	99999PR	–3040	<3040>
EEEE	Scientific notation	99.99999EEEE	121.976	1.21976E+02
U	nls_dual_currency	U099999	03040	CAD003040 if nls_dual_currency is set to CAD
V	Multiplies by 10n times (n is the number of nines after V)	9999V99	3040	304000
S	+ or – sign is prefixed	S999999	3040	+3040

QUESTION 6
Examine the structure of the SHIPMENTS table:

name Null Type
PO_ID NOT NULL NUMBER(3)
PO_DATE NOT NULL DATE
SHIPMENT_DATE NOT NULL DATE
SHIPMENT_MODE VARCHAR2(30)
SHIPMENT_COST NUMBER(8,2)

You want to generate a report that displays the PO_ID and the penalty amount to be paid if the SHIPMENT_DATE is later than one month from the PO_DATE. The penalty is $20 per day.
Evaluate the following two queries:

```
SQL> SELECT po_id, CASE
WHEN MONTHS_BETWEEN (shipment_date,po_date)>1 THEN
TO_CHAR((shipment_date - po_date) * 20) ELSE 'No Penalty' END PENALTY
FROM shipments;

SQL>SELECT po_id, DECODE
(MONTHS_BETWEEN (po_date,shipment_date)>1,
TO_CHAR((shipment_date - po_date) * 20), 'No Penalty') PENALTY
FROM shipments;
```

Which statement is true regarding the above commands?

A. Both execute successfully and give correct results.
B. Only the first query executes successfully but gives a wrong result.
C. Only the first query executes successfully and gives the correct result.
D. Only the second query executes successfully but gives a wrong result.
E. Only the second query executes successfully and gives the correct result.

Answer: C
Section: (none)

Explanation/Reference:
The **MONTHS_BETWEEN**(date 1, date 2) function returns the number of
months between two dates:
months_between('01-FEB-2008','01-JAN-2008') = 1

The DECODE Function

Although its name sounds mysterious, this function is straightforward. The DECODE function implements if-then-else conditional logic by testing its first two terms for equality and returns the third if they are equal and optionally returns another term if they are not.

DECODE Function

Facilitates conditional inquiries by doing the work of a CASE expression or an IF-THEN-ELSE statement:

DECODE(col|expression, search1, result1
[, search2, result2,...,]
[, default])

DECODE Function

The DECODE function decodes an expression in a way similar to the IF-THEN-ELSE logic that is used in various languages. The DECODE function decodes expression after comparing it to each search value. If the expression is the same as search, result is returned.
If the default value is omitted, a null value is returned where a search value does not match anyof the result values

QUESTION 7
Which two statements are true regarding the USING and ON clauses in table joins? (Choose two.)

A. Both USING and ON clauses can be used for equijoins and nonequijoins.
B. A maximum of one pair of columns can be joined between two tables using the ON clause.
C. The ON clause can be used to join tables on columns that have different names but compatible data types.
D. The WHERE clause can be used to apply additional conditions in SELECT statements containing the ON or the USING clause.

Answer: CD
Section: (none)

Explanation/Reference:

Creating Joins with the USING Clause

If several columns have the same names but the data types do not match, use the USING clause to specify the columns for the equijoin.
Use the USING clause to match only one column when more than one column matches.
The NATURAL JOIN and USING clauses are mutually exclusive

Using Table Aliases with the USING clause

When joining with the USING clause, you cannot qualify a column that is used in the USING clause itself. Furthermore, if that column is used anywhere in the SQL statement, you cannot alias it. For example, in the query mentioned in the slide, you should not alias the location_id column in the WHERE clause because the column is used in the USING clause.
The columns that are referenced in the USING clause should not have a qualifier (table name oralias) anywhere in the SQL statement.

Creating Joins with the ON Clause

The join condition for the natural join is basically an equijoin of all columns with the same name.
Use the ON clause to specify arbitrary conditions or specify columns to join. – ANSWER C
The join condition is separated from other search conditions. ANSWER D
The ON clause makes code easy to understand.

QUESTION 8
View the Exhibit and examine the structure of the CUSTOMERS table.

Which two tasks would require subqueries or joins to be executed in a single statement? (Choose two.)

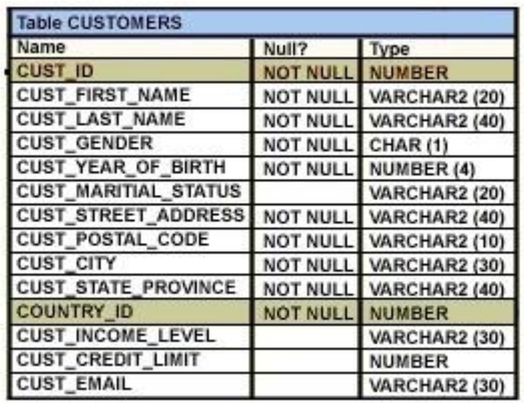


Table CUSTOMERS		
Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

A. listing of customers who do not have a credit limit and were born before 1980
B. finding the number of customers, in each city, whose marital status is 'married'
C. finding the average credit limit of male customers residing in 'Tokyo' or 'Sydney'
D. listing of those customers whose credit limit is the same as the credit limit of customers residing in the city 'Tokyo'
E. finding the number of customers, in each city, whose credit limit is more than the average credit limit of all the customers

Answer: DE
Section: (none)

Explanation/Reference:

Describe the Types of Problems That the Subqueries Can Solve

There are many situations where you will need the result of one query as the input for another.

Use of a Subquery Result Set for Comparison Purposes

Which employees have a salary that is less than the average salary? This could be answered by two statements, or by a single statement with a subquery. The following example uses two statements:
select avg(salary) from employees;
select last_name from employees where salary < result_of_previous_query ;
Alternatively, this example uses one statement with a subquery:
select last_name from employees
where salary < (select avg(salary)from employees);
In this example, the subquery is used to substitute a value into the WHERE clause of the parent query: it is returning a single value, used for comparison with the rows retrieved by the parent query.
The subquery could return a set of rows. For example, you could use the following to find all departments that do actually have one or more employees assigned to them:
select department_name from departments where department_id in
(select distinct(department_id) from employees);

QUESTION 9
Which statement is true regarding the INTERSECT operator?

A. It ignores NULL values.
B. Reversing the order of the intersected tables alters the result.
C. The names of columns in all SELECT statements must be identical.
D. The number of columns and data types must be identical for all SELECT statements in the query.

Answer: D
Section: (none)

Explanation/Reference:
INTERSECT Returns only the rows that occur in both queries' result sets, sorting them and removing duplicates.

The columns in the queries that make up a compound query can have different names, but the output result set will use the names of the columns in the first query

QUESTION 10
View the Exhibit; e xamine the structure of the PROMOTIONS table.

Each promotion has a duration of at least seven days .

Your manager has asked you to generate a report, which provides the weekly cost for each promotion done to l date.

Which query would achieve the required result?

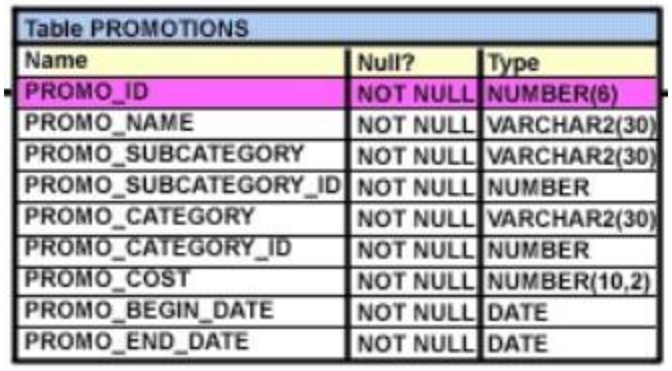


Table PROMOTIONS		
Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

A. SELECT promo_name, promo_cost/promo_end_date-promo_begin_date/7 FROM promotions;
B. SELECT promo_name,(promo_cost/promo_end_date-promo_begin_date)/7 FROM promotions;
C. SELECT promo_name, promo_cost/(promo_end_date-promo_begin_date/7) FROM promotions;
D. SELECT promo_name, promo_cost/((promo_end_date-promo_begin_date)/7) FROM promotions;

Answer: D
Section: (none)

Explanation/Reference:

QUESTION 11
View the Exhibit and examine the structure of the PRODUCTS table.

All products have a list price.

You issue the following command to display the total price of each product after a discount of 25% and a tax of 15% are applied on it. Freight charges of $100 have to be applied to all the products.

```
SQL>SELECT prod_name, prod_list_price -(prod_list_price*(25/100))
+(prod_list_price -(prod_list_price*(25/100))*(15/100))+100
```

AS "TOTAL PRICE"
FROM products;

What would be the outcome if all the parenthese s are removed from the above statement?

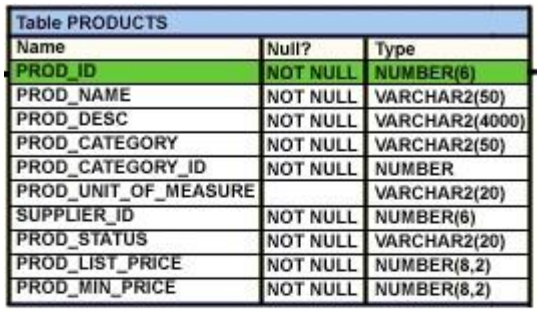


Table PRODUCTS		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

A. It produces a syntax error.
B. The result remains unchanged.
C. The total price value would be lower than the correct value.
D. The total price value would be higher than the correct value.

Answer: B
Section: (none)

Explanation/Reference:

QUESTION 12
You need to produce a report where each customer's credit limit has been incremented by $1000.
In the output, the customer's last name should have the heading Name and the incremented credit limit should be labeled New Credit Limit.

The column headings should have only the first letter of each word in uppercase .

Which statement would accomplish this requirement?

A. SELECT cust_last_name Name, cust_credit_limit + 1000
"New Credit Limit"
FROM customers;
B. SELECT cust_last_name AS Name, cust_credit_limit + 1000
AS New Credit Limit
FROM customers;
C. SELECT cust_last_name AS "Name", cust_credit_limit + 1000
AS "New Credit Limit"
FROM customers;
D. SELECT INITCAP(cust_last_name) "Name", cust_credit_limit + 1000 INITCAP("NEW CREDIT LIMIT")
FROM customers;

Answer: C
Section: (none)

Explanation/Reference:
A column alias:
- Renames a column heading
- Is useful with calculations
- Immediately follows the column name (There can also be the optional AS keyword between the column name and the alias.)

- Requires double quotation marks if it contains spaces or special characters, or if it is case-sensitive

QUESTION 13
View the Exhibit and examine the structure of the PRODUCTS table.

You need to generate a report in the following format:

CATEGORIES

5MP Digital Photo Camera's category is Photo
Y Box's category is Electronics

Envoy Ambassador's category is Hardware

Which two queries would give the required output? (Choose two.)

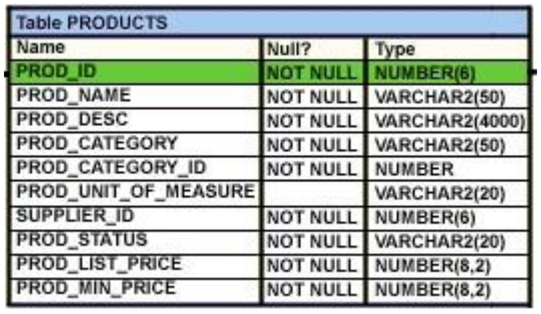


Table PRODUCTS		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

A. SELECT prod_name || q'''s category is ' || prod_category CATEGORIES FROM products;
B. SELECT prod_name || q'['s]'category is ' || prod_category CATEGORIES FROM products;
C. SELECT prod_name || q'\'s\' || ' category is ' || prod_category CATEGORIES FROM products;
D. SELECT prod_name || q'<'s >' || 'category is ' || prod_category CATEGORIES FROM products;

Answer: CD
Section: (none)

Explanation/Reference:
So, how are words that contain single quotation marks dealt with? There are essentially two mechanisms available. The most popular of these is to add **an additional single quotation mark** next to each naturally occurring single quotation mark in the character string
Oracle offers a neat way to deal with this type of character literal in the form of the alternative **quote (q) operator**. Notice that the problem is that Oracle chose the single quote characters as the special pair of symbols that enclose or wrap any other character literal. These character-enclosing symbols could have been anything other than single quotation marks.
Bearing this in mind, consider the alternative quote (q) operator. The q operator enables you to choose from a set of possible pairs of wrapping symbols for character literals as alternatives to the single quote symbols. The options are any single-byte or multibyte character or the four brackets: (round brackets), {curly braces}, [squarebrackets], or <angle brackets>. Using the q operator, the character delimiter can effectively be changed from a single quotation mark to any other character
The syntax of the alternative quote operator is as follows:
q'delimiter'character literal which may include the single quotes delimiter' where delimiter can be any character or bracket.

Alternative Quote (q) Operator

Specify your own quotation mark delimiter.
Select any delimiter.
Increase readability and usability.

SELECT department_name || q'[Department's Manager Id:]'
|| manager_id

AS "Department and Manager"
FROM departments;

Alternative Quote (q) Operator

Many SQL statements use character literals in expressions or conditions. If the literal itself contains a single quotation mark, you can use the quote (q) operator and select your own quotation mark delimiter.
You can choose any convenient delimiter, single-byte or multibyte, or any of the following character pairs: [ ], { }, ( ), or < >.
In the example shown, the string contains a single quotation mark, which is normally interpreted as a delimiter of a character string. By using the q operator, however, brackets [] are used as the quotation mark delimiters. The string between the brackets delimiters is interpreted as a literal character string.

QUESTION 14
Using the CUSTOMERS table, you need to generate a report that shows 50% of each credit amount in each income level. The report should NOT show any repeated credit amounts in each income level.

Which query would give the required result?

A. SELECT cust_income_level, DISTINCT cust_credit_limit * 0.50 AS "50% Credit Limit"
   FROM customers;
B. SELECT DISTINCT cust_income_level, DISTINCT cust_credit_limit * 0.50 AS "50% Credit Limit"
   FROM customers;
C. SELECT DISTINCT cust_income_level ' ' cust_credit_limit * 0.50 AS "50% Credit Limit"
   FROM customers;
D. SELECT cust_income_level ' ' cust_credit_limit * 0.50 AS "50% Credit Limit" FROM customers;

Answer: C
Section: (none)

Explanation/Reference:

Duplicate Rows

Unless you indicate otherwise, SQL displays the results of a query without eliminating the duplicate rows.
To eliminate duplicate rows in the result, include the **DISTINCT** keyword in the SELECT clause immediately after the SELECT keyword.
You can specify multiple columns after the DISTINCT qualifier. The DISTINCT qualifier affects all the selected columns, and the result is every distinct combination of the columns.

QUESTION 15
View the Exhibit and examine the data in the CUSTOMERS table.

Evaluate the following query:

```
SQL> SELECT cust_name AS "NAME", cust_credit_limit/2 AS MIDPOINT,MIDPOINT+100 AS "MAX LOWER
LIMIT"
FROM customers;
```

The above query produces an error on execution.
What is the reason for the error?

A. An alias cannot be used in an expression.
B. The a lias NAME should not be enclosed with in double quotation marks .
C. The MIDPOINT+100 expression gives an error because CUST_CREDIT_LIMIT contains NULL values.

D. The a lias MIDPOINT should be enclosed with in double quotation marks for the CUST_CREDIT_LIMIT/2 expression .

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 16
Evaluate the following query:

```
SQL> SELECT promo_name || q'{'s start date was }' || promo_begin_date
AS "Promotion Launches"
FROM promotions;
```

What would be the outcome of the above query?

A. It produces an error because flower braces have been used.
B. It produces an error because the data types are not matching.
C. It executes successfully and introduces an 's at the end of each promo_name in the output.
D. It executes successfully and displays the literal " {'s start date was } " for each row in the output.

Answer: C
Section: (none)

Explanation/Reference:
So, how are words that contain single quotation marks dealt with? There are essentially two mechanisms available. The most popular of these is to add **an additional single quotation mark** next to each naturally occurring single quotation mark in the character string
Oracle offers a neat way to deal with this type of character literal in the form of the alternative **quote (q) operator**. Notice that the problem is that Oracle chose the single quote characters as the special pair of symbols that enclose or wrap any other character literal. These character-enclosing symbols could have been anything other than single quotation marks.
Bearing this in mind, consider the alternative quote (q) operator. The q operator enables you to choose from a set of possible pairs of wrapping symbols for character literals as alternatives to the single quote symbols. The options are any single-byte or multibyte character or the four brackets: (round brackets), {curly braces}, [squarebrackets], or <angle brackets>. Using the q operator, the character delimiter can effectively be changed from a single quotation mark to any other character
The syntax of the alternative quote operator is as follows:
q'delimiter'character literal which may include the single quotes delimiter' where delimiter can be any character or bracket.

Alternative Quote (q) Operator

Specify your own quotation mark delimiter.
Select any delimiter.
Increase readability and usability.

```
SELECT department_name || q'[ Department's Manager Id: ]'
|| manager_id
AS "Department and Manager"
FROM departments;
```

Alternative Quote (q) Operator

Many SQL statements use character literals in expressions or conditions. If the literal itself contains a single quotation mark, you can use the quote (q) operator and select your own quotation mark delimiter.
You can choose any convenient delimiter, single-byte or multibyte, or any of the following character pairs: [ ], { }, ( ), or < >.
In the example shown, the string contains a single quotation mark, which is normally interpreted as a delimiter of a character string. By using the q operator, however, brackets [] are used as the quotation mark delimiters. The string between the brackets delimiters is interpreted as a literal character string.

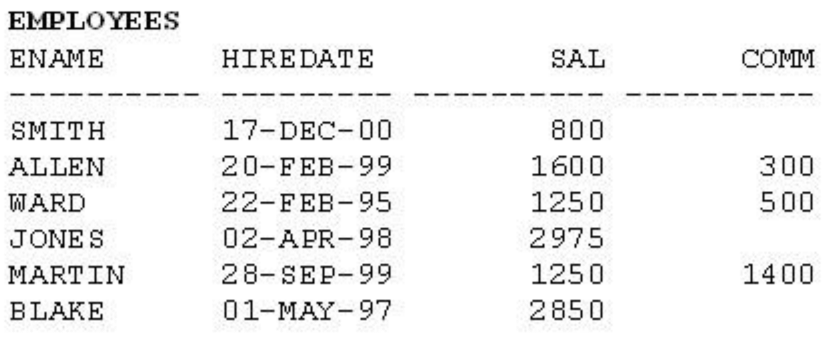
QUESTION 17
View the Exhibit and examine the data in the EMPLOYEES table.

You want to generate a report showing the total compensation paid to each employee to date.

You issue the following query:

```
SQL>SELECT ename ' joined on ' hiredate
', the total compensation paid is '
TO_CHAR(ROUND(ROUND(SYSDATE-hiredate)/365) * sal + comm)
"COMPENSATION UNTIL DATE"
FROM employees;
```

What is the outcome?



EMPLOYEES

ENAME	HIREDATE	SAL	COMM
SMITH	17-DEC-00	800	
ALLEN	20-FEB-99	1600	300
WARD	22-FEB-95	1250	500
JONES	02-APR-98	2975	
MARTIN	28-SEP-99	1250	1400
BLAKE	01-MAY-97	2850	

A. It generates an error because the alias is not valid.
B. It executes successfully and gives the correct output.
C. It executes successfully but does not give the correct output.
D. It generates an error because the usage of the ROUND function in the expression is not valid.
E. It generates an error because the concatenation operator can be used to combine only two items.

Answer: C
Section: (none)

Explanation/Reference:
ROUND(**column|expression**, **n**) Rounds the column, expression, or value to **n** decimal places or, if **n** is omitted, no decimal places (If **n** is negative, numbers to the left of decimal point are rounded.)

QUESTION 18
Examine the structure of the PROMOTIONS table:

```
name Null Type
PROMO_ID NOT NULL NUMBER(6)
PROMO_NAME NOT NULL VARCHAR2(30)
PROMO_CATEGORY NOT NULL VARCHAR2(30)
PROMO_COST NOT NULL NUMBER(10,2)
```

The management wants to see a report of unique promotion costs in each promotion category.

Which query would achieve the required result?

A. SELECT DISTINCT promo_cost, promo_category FROM promotions;
B. SELECT promo_category, DISTINCT promo_cost FROM promotions;

C. SELECT DISTINCT promo_cost, DISTINCT promo_category FROM promotions;
D. SELECT DISTINCT promo_category, promo_cost FROM promotions ORDER BY 1;

Answer: D
Section: (none)

Explanation/Reference:

QUESTION 19
Evaluate the following query:

```
SELECT INTERVAL '300' MONTH,
INTERVAL '54-2' YEAR TO MONTH,
INTERVAL '11:12:10.1234567' HOUR TO SECOND
FROM dual;
```

What is the correct output of the above query?

A. +25-00 , +54-02, +00 11:12:10.123457
B. +00-300, +54-02, +00 11:12:10.123457
C. +25-00 , +00-650, +00 11:12:10.123457
D. +00-300 , +00-650, +00 11:12:10.123457

Answer: A
Section: (none)

Explanation/Reference:
Datetime Data Types
You can use several datetime data types:
INTERVAL YEAR TO MONTH
Stored as an interval of **years and months**
INTERVAL DAY TO SECOND
Stored as an interval of **days, hours, minutes,and seconds**

QUESTION 20
Which three statements are true regarding the data types in Oracle Database 10g/11g? (Choose three.)

A. Only one LONG column can be used per table.
B. A TIMESTAMP data type column stores only time values with fractional seconds.
C. The BLOB data type column is used to store binary data in an operating system file.
D. The minimum column width that can be specified for a VARCHAR2 data type column is one.
E. The value for a CHAR data type column is blank-padded to the maximum defined column width.

Answer: ADE
Section: (none)

Explanation/Reference:
■ **LONG** Character data in the database character set, up to 2GB. All the functionality of LONG (and more) is provided by CLOB; LONGs should not be used in a modern database, and if your database has any columns of this type they should be converted to CLOB. There can **only be one** LONG column in a table.
DVARCHAR2 Variable-length character data, from 1 byte to 4KB. The data is stored in the database character set. The VARCHAR2 data type must be qualified with a number indicating the maximum length of the column. If a value is inserted into the column that is less than this, it is not a problem: the value will only take up as much space as it needs. If the value is longer than this maximum, the INSERT will fail with an error.
VARCHAR2(**size**)
Variable-length character data (A maximum **size** must be specified: minimum **size** is 1; maximum **size** is 4,000.)

BLOB Like CLOB, but binary data that will not undergo character set conversion by Oracle Net.
BFILE A locator pointing to a file stored on the operating system of the database server. The size of the files is limited to 4GB.
TIMESTAMP This is length zero if the column is empty, or up to 11 bytes, depending on the precision specified. Similar to DATE, but with precision of up to 9 decimal places for the seconds, 6 places by default.

QUESTION 21
Examine the description of the EMP_DETAILS table given below:

name NULL TYPE
EMP_ID NOT NULL NUMBER
EMP_NAME NOT NULL VARCHAR2 (40)
EMP_IMAGE LONG

Which two statements are true regarding SQL statements that can be executed on the EMP_DETAIL table? (Choose two.)

A. An EMP_IMAGE column can be included in the GROUP BY clause.
B. An EMP_IMAGE column cannot be included in the ORDER BY clause.
C. You cannot add a new column to the table with LONG as the data type.
D. You can alter the table to include the NOT NULL constraint on the EMP_IMAGE column.

Answer: BC
Section: (none)

Explanation/Reference:
LONG Character data in the database character set, up to 2GB. All the functionality of LONG (and more) is provided by CLOB; LONGs should not be used in a modern database, and if your database has any columns of this type they should be converted to CLOB. **There can only be one** LONG column in a table.

Guidelines

A LONG column is not copied when a table is created using a subquery.
A LONG column cannot be included in a GROUP BY or an ORDER BY clause.
Only one LONG column can be used per table.
No constraints can be defined on a LONG column.
You might want to use a CLOB column rather than a LONG column.

QUESTION 22
You need to create a table for a banking application. One of the columns in the table has the following requirements:

1) You want a column in the table to store the duration of the credit period.
2) The data in the column should be stored in a format such that it can be easily added and subtracted with DATE data type without using conversion functions.
3) The maximum period of the credit provision in the application is 30 days.
4) The interest has to be calculated for the number of days an individual has taken a credit for.

Which data type would you use for such a column in the table?

A. DATE
B. NUMBER
C. TIMESTAMP
D. INTERVAL DAY TO SECOND
E. INTERVAL YEAR TO MONTH

Answer: D
Section: (none)

Explanation/Reference:

QUESTION 23
Examine the structure proposed for the TRANSACTIONS table:

name Null Type
TRANS_ID NOT NULL NUMBER(6)
CUST_NAME NOT NULL VARCHAR2(20)
CUST_STATUS NOT NULL CHAR
TRANS_DATE NOT NULL DATE
TRANS_VALIDITY VARCHAR2
CUST_CREDIT_LIMIT NUMBER

Which statements are true regarding the creation and storage of data in the above table structure? (Choose all that apply.)

A. The CUST_STATUS column would give an error.
B. The TRANS_VALIDITY column would give an error.
C. The CUST_STATUS column would store exactly one character.
D. The CUST_CREDIT_LIMIT column would not be able to store decimal values.
E. The TRANS_VALIDITY column would have a maximum size of one character.
F. The TRANS_DATE column would be able to store day, month, century, year, hour, minutes, seconds, and fractions of seconds.

Answer: BC
Section: (none)

Explanation/Reference:
VARCHAR2(size) Variable-length character data (A maximum **size** must be specified: minimum **size** is 1; maximum **size** is 4,000.)
CHAR [(size)] Fixed-length character data of length **size** bytes (Default and minimum **size** is 1; maximum **size** is 2,000.)
NUMBER [(p,s)] Number having precision **p** and scale **s** (Precision is the total number of decimal digits and scale is the number of digits to the right of the decimal point; precision can range from 1 to 38, and scale can range from –84 to 127.)
DATE Date and time values to the nearest second between January 1, 4712 B.C., and December 31, 9999 A.D.

QUESTION 24
Examine the structure proposed for the TRANSACTIONS table:

name Null Type
TRANS_ID NOT NULL NUMBER(6)
CUST_NAME NOT NULL VARCHAR2(20)
CUST_STATUS NOT NULL VARCHAR2
TRANS_DATE NOT NULL DATE
TRANS_VALIDITY INTERVAL DAY TO SECOND
CUST_CREDIT_VALUE NUMBER(10)

Which two statements are true regarding the storage of data in the above table structure? (Choose two.)

A. The TRANS_DATE column would allow storage of dates only in the dd-mon-yyyy format.
B. The CUST_CREDIT_VALUE column would allow storage of positive and negative integers.
C. The TRANS_VALIDITY column would allow storage of a time interval in days, hours, minutes, and seconds.
D. The CUST_STATUS column would allow storage of data up to the maximum VARCHAR2 size of 4,000 characters.

Answer: BC
Section: (none)

Explanation/Reference:

QUESTION 25
Resume (character large object [CLOB] data type), which contains the resume submitted by the employee

Which is the correct syntax to create this table?

A. CREATE TABLE EMP_1
(emp_id NUMBER(4),
emp_name VARCHAR2(25),
start_date DATE,
e_status VARCHAR2(10) DEFAULT 'ACTIVE',
resume CLOB(200));
B. CREATE TABLE 1_EMP
(emp_id NUMBER(4),
emp_name VARCHAR2(25),
start_date DATE,
emp_status VARCHAR2(10) DEFAULT 'ACTIVE',
resume CLOB);
C. CREATE TABLE EMP_1
(emp_id NUMBER(4),
emp_name VARCHAR2(25),
start_date DATE,
emp_status VARCHAR2(10) DEFAULT "ACTIVE",
resume CLOB);
D. CREATE TABLE EMP_1
(emp_id NUMBER,
emp_name VARCHAR2(25),
start_date DATE,
emp_status VARCHAR2(10) DEFAULT 'ACTIVE',
resume CLOB);

Answer: D
Section: (none)

Explanation/Reference:
CLOB Character data (up to 4 GB)
NUMBER [(**p,s**)] Number having precision **p** and scale **s** (Precision is the total number of decimal digits and scale is the number of digits to the right of the decimal point; precision can range from 1 to 38, and scale can range from –84 to 127.)

QUESTION 26
Which is the valid CREATE TABLE statement?

A. CREATE TABLE emp9$# (emp_no NUMBER (4));
B. CREATE TABLE 9emp$# (emp_no NUMBER(4));
C. CREATE TABLE emp*123 (emp_no NUMBER(4));
D. CREATE TABLE emp9$# (emp_no NUMBER(4), date DATE);

Answer: A
Section: (none)

Explanation/Reference:
Schema Object Naming Rules
Every database object has a name. In a SQL statement, you represent the name of an object with a **quoted identifier** or a **nonquoted identifier**.

A quoted identifier begins and ends with **double quotation marks (").** If you name a schema object using a quoted identifier, then you must use the double quotation marks whenever you refer to that object.
A nonquoted identifier is not surrounded by any punctuation.
The following list of rules applies to both quoted and nonquoted identifiers unless otherwise indicated:
Names must be from 1 to 30 bytes long with these exceptions:
Names of databases are limited to 8 bytes.
Names of database links can be as long as 128 bytes.
If an identifier includes multiple parts separated by periods, then each attribute can be up to 30 bytes long. Each period separator, as well as any surrounding double quotation marks, counts as one byte. For example, suppose you identify a column like this:
"**schema**"."**table**"."**column**"

Nonquoted identifiers cannot be Oracle Database **reserved words (ANSWER D)**. Quoted identifiers can be reserved words, although this is not recommended.
Depending on the Oracle product you plan to use to access a database object, names might be further restricted by other product-specific reserved words.
The Oracle SQL language contains other words that have special meanings. These words include datatypes, schema names, function names, the dummy system table DUAL, and keywords (the uppercase words in SQL statements, such as DIMENSION, SEGMENT, ALLOCATE, DISABLE, and so forth). These words are not reserved. However, Oracle uses them internally in specific ways. Therefore, if you use these words as names for objects and object parts, then your SQL statements may be more difficult to read and may lead to unpredictable results.
In particular, do not use words beginning with SYS_ as schema object names, and do not use the names of SQL built-in functions for the names of schema objects or user-defined functions.
You should use ASCII characters in database names, global database names, and database link names, because ASCII characters provide optimal compatibility across different platforms and operating systems.
Nonquoted identifiers must **begin with an alphabetic character** (ANSWER B - begins with 9) from your database character set. Quoted identifiers can begin with any character.
Nonquoted identifiers can contain only alphanumeric characters from your database character set and the underscore (_), dollar sign ($), and pound sign (#). Database links can also contain periods (.) and "at" signs (@). Oracle strongly discourages you from using $ and # in nonquoted identifiers.
Quoted identifiers can contain any characters and punctuations marks as well as spaces. However, neither quoted nor nonquoted identifiers can contain double quotation marks or the null character (\0).
Within a namespace, no two objects can have **the same name**.
Nonquoted identifiers are **not case sensitive**. Oracle interprets them as uppercase. Quoted identifiers are case sensitive.
By enclosing names in double quotation marks, you can give the following names to different objects in the same namespace:
employees
"employees"
"Employees"
"EMPLOYEES"

Note that Oracle interprets the following names the same, so they cannot be used for different objects in the same namespace:
employees
EMPLOYEES
"EMPLOYEES"

Columns in the same table or view cannot have the same name. However, columns in different tables or views can have the same name.
Procedures or functions contained in the same package can have the same name, if their arguments are not of the same number and datatypes. Creating multiple procedures or functions with the same name in the same package with different arguments is called **overloading** the procedure or function.

QUESTION 27
Which two statements are true regarding tables? (Choose two.)

A. A table name can be of any length.
B. A table can have any number of columns.
C. A column that has a DEFAULT value cannot store null values.
D. A table and a view can have the same name in the same schema.

E. A table and a synonym can have the same name in the same schema.
F. The same table name can be used in different schemas in the same database.

Answer: EF
Section: (none)

Explanation/Reference:

Synonyms
Synonyms are database objects that enable you to call a table by another name. You can create synonyms to give an alternative name to a table.

QUESTION 28
Which two statements are true regarding constraints? (Choose two.)

A. A foreign key cannot contain NULL values.
B. A column with the UNIQUE constraint can contain NULL values.
C. A constraint is enforced only for the INSERT operation on a table.
D. A constraint can be disabled even if the constraint column contains data.
E. All constraints can be defined at the column level as well as the table level.

Answer: BD
Section: (none)

Explanation/Reference:
Including Constraints
• Constraints enforce rules at the table level.
• Constraints prevent the deletion of a table if there are dependencies.
• The following constraint types are valid:
– NOT NULL
– UNIQUE
– PRIMARY KEY
– FOREIGN KEY
– CHECK

QUESTION 29
Which two statements are true regarding constraints? (Choose two.)

A. A foreign key cannot contain NULL values.
B. The column with a UNIQUE constraint can store NULLS .
C. A constraint is enforced only for an INSERT operation on a table.
D. You can have more than one column in a table as part of a primary key.

Answer: BD
Section: (none)

Explanation/Reference:

QUESTION 30
Evaluate the following CREATE TABLE commands:

```
CREATE TABLE orders
(ord_no NUMBER(2) CONSTRAINT ord_pk PRIMARY KEY,
ord_date DATE,
cust_id NUMBER(4));
CREATE TABLE ord_items
(ord_no NUMBER(2),
item_no NUMBER(3),
qty NUMBER(3) CHECK (qty BETWEEN 100 AND 200),
expiry_date date CHECK (expiry_date > SYSDATE),
CONSTRAINT it_pk PRIMARY KEY (ord_no,item_no),
CONSTRAINT ord_fk FOREIGN KEY(ord_no) REFERENCES orders(ord_no));
```

The above command fails when executed. What could be the reason?

A. SYSDATE cannot be used with the CHECK constraint.
B. The BETWEEN clause cannot be used for the CHECK constraint.
C. The CHECK constraint cannot be placed on columns having the DATE data type.
D. ORD_NO and ITEM_NO cannot be used as a composite primary key because ORD_NO is also the FOREIGN KEY.

Answer: A
Section: (none)

Explanation/Reference:
CHECK Constraint
The CHECK constraint defines a condition that each row must satisfy. The condition can use the same constructs as the query conditions, with the following exceptions:
 References to the CURRVAL, NEXTVAL, LEVEL, and ROWNUM pseudocolumns
 Calls to SYSDATE, UID, USER, and USERENV functions
 Queries that refer to other values in other rows
A single column can have multiple CHECK constraints that refer to the column in its definition.
There is no limit to the number of CHECK constraints that you can define on a column.
CHECK constraints can be defined at the column level or table level.
CREATE TABLE employees
(...
salary NUMBER(8,2) CONSTRAINT emp_salary_min
CHECK (salary > 0),

QUESTION 31
Evaluate the following SQL commands:

```
SQL>CREATE SEQUENCE ord_seq
INCREMENT BY 10
START WITH 120
MAXVALUE 9999
NOCYCLE;

SQL>CREATE TABLE ord_items
(ord_no NUMBER(4)DEFAULT ord_seq.NEXTVAL NOT NULL,
item_no NUMBER(3),
qty NUMBER(3) CHECK (qty BETWEEN 100 AND 200),
expiry_date date CHECK (expiry_date > SYSDATE),
CONSTRAINT it_pk PRIMARY KEY (ord_no,item_no),
CONSTRAINT ord_fk FOREIGN KEY(ord_no) REFERENCES orders(ord_no));
```

The command to create a table fails. Identify the reason for the SQL statement failure? (Choose all that apply.)

A. You cannot use SYSDATE in the condition of a CHECK constraint.
B. You cannot use the BETWEEN clause in the condition of a CHECK constraint.
C. You cannot use the NEXTVAL sequence value as a DEFAULT value for a column.
D. You cannot use ORD_NO and ITEM_NO columns as a composite primary key because ORD_NO is also the FOREIGN KEY.

Answer: AC
Section: (none)

Explanation/Reference:
CHECK Constraint
The CHECK constraint defines a condition that each row must satisfy. The condition can use the same constructs as the query conditions, with the following exceptions:
 References to the CURRVAL, NEXTVAL, LEVEL, and ROWNUM pseudocolumns
 Calls to SYSDATE, UID, USER, and USERENV functions
 Queries that refer to other values in other rows
A single column can have multiple CHECK constraints that refer to the column in its definition.
There is no limit to the number of CHECK constraints that you can define on a column.
CHECK constraints can be defined at the column level or table level.
CREATE TABLE employees
(...
salary NUMBER(8,2) CONSTRAINT emp_salary_min
CHECK (salary > 0),

QUESTION 32
Which CREATE TABLE statement is valid?

A. CREATE TABLE ord_details
(ord_no NUMBER(2) PRIMARY KEY,
item_no NUMBER(3) PRIMARY KEY,
ord_date DATE NOT NULL);
B. CREATE TABLE ord_details
(ord_no NUMBER(2) UNIQUE, NOT NULL,
item_no NUMBER(3),
ord_date DATE DEFAULT SYSDATE NOT NULL);
C. CREATE TABLE ord_details
(ord_no NUMBER(2) ,
item_no NUMBER(3),
ord_date DATE DEFAULT NOT NULL,
CONSTRAINT ord_uq UNIQUE (ord_no),
CONSTRAINT ord_pk PRIMARY KEY (ord_no));
D. CREATE TABLE ord_details
(ord_no NUMBER(2),
item_no NUMBER(3),
ord_date DATE DEFAULT SYSDATE NOT NULL,
CONSTRAINT ord_pk PRIMARY KEY (ord_no, item_no));

Answer: D
Section: (none)

Explanation/Reference:
PRIMARY KEY Constraint
A PRIMARY KEY constraint creates a primary key for the table. Only one primary key can be created for each table. The PRIMARY KEY constraint is a column or a set of columns that uniquely identifies each row in a table. This constraint enforces the uniqueness of the column or column combination and ensures that no column that is part of the primary key can contain a null value.
Note: Because uniqueness is part of the primary key constraint definition, the Oracle server enforces the uniqueness by implicitly creating a unique index on the primary key column or columns.

QUESTION 33

You want to create an ORD_DETAIL table to store details for an order placed having the following business requirement:

1) The order ID will be unique and cannot have null values.
2) The order date cannot have null values and the default should be the current date.
3) The order amount should not be less than 50.
4) The order status will have values either shipped or not shipped.
5) The order payment mode should be cheque, credit card, or cash on delivery (COD).

Which is the valid DDL statement for creating the ORD_DETAIL table?

A. CREATE TABLE ord_details
(ord_id NUMBER(2) CONSTRAINT ord_id_nn NOT NULL,
ord_date DATE DEFAULT SYSDATE NOT NULL,
ord_amount NUMBER(5, 2) CONSTRAINT ord_amount_min
CHECK (ord_amount > 50),
ord_status VARCHAR2(15) CONSTRAINT ord_status_chk
CHECK (ord_status IN ('Shipped', 'Not Shipped')),
ord_pay_mode VARCHAR2(15) CONSTRAINT ord_pay_chk
CHECK (ord_pay_mode IN ('Cheque', 'Credit Card',
'Cash On Delivery')));

B. CREATE TABLE ord_details
(ord_id NUMBER(2) CONSTRAINT ord_id_uk UNIQUE NOT NULL,
ord_date DATE DEFAULT SYSDATE NOT NULL,
ord_amount NUMBER(5, 2) CONSTRAINT ord_amount_min
CHECK (ord_amount > 50),
ord_status VARCHAR2(15) CONSTRAINT ord_status_chk
CHECK (ord_status IN ('Shipped', 'Not Shipped')),
ord_pay_mode VARCHAR2(15) CONSTRAINT ord_pay_chk
CHECK (ord_pay_mode IN ('Cheque', 'Credit Card',
'Cash On Delivery')));

C. CREATE TABLE ord_details
(ord_id NUMBER(2) CONSTRAINT ord_id_pk PRIMARY KEY,
ord_date DATE DEFAULT SYSDATE NOT NULL,
ord_amount NUMBER(5, 2) CONSTRAINT ord_amount_min
CHECK (ord_amount >= 50),
ord_status VARCHAR2(15) CONSTRAINT ord_status_chk
CHECK (ord_status IN ('Shipped', 'Not Shipped')),
ord_pay_mode VARCHAR2(15) CONSTRAINT ord_pay_chk
CHECK (ord_pay_mode IN ('Cheque', 'Credit Card',
'Cash On Delivery')));

D. CREATE TABLE ord_details
(ord_id NUMBER(2),
ord_date DATE NOT NULL DEFAULT SYSDATE,
ord_amount NUMBER(5, 2) CONSTRAINT ord_amount_min
CHECK (ord_amount >= 50),
ord_status VARCHAR2(15) CONSTRAINT ord_status_chk
CHECK (ord_status IN ('Shipped', 'Not Shipped')),
ord_pay_mode VARCHAR2(15) CONSTRAINT ord_pay_chk
CHECK (ord_pay_mode IN ('Cheque', 'Credit Card',
'Cash On Delivery')));

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 34
You created an ORDERS table with the following description:

```
name Null Type
ORD_ID NOT NULL NUMBER(2)
CUST_ID NOT NULL NUMBER(3)
ORD_DATE NOT NULL DATE
ORD_AMOUNT NOT NULL NUMBER (10,2)
```

You inserted some rows in the table. After some time, you want to alter the table by creating the PRIMARY KEY constraint on the ORD_ID column. Which statement is true in this scenario?

A. You cannot have two constraints on one column.
B. You cannot add a primary key constraint if data exists in the column.
C. The primary key constraint can be created only at the time of table creation .
D. You can add the primary key constraint even if data exists, provided that there are no duplicate values.

Answer: D
Section: (none)

Explanation/Reference:

QUESTION 35
Which two statements are true regarding constraints? (Choose two.)

A. A table can have only one primary key and one foreign key.
B. A table can have only one primary key but multiple foreign keys.
C. Only the primary key can be defined at the column and table levels.
D. The foreign key and parent table primary key must have the same name.
E. Both primary key and foreign key constraints can be defined at both column and table levels.

Answer: BE
Section: (none)

Explanation/Reference:

QUESTION 36
Examine the following SQL commands:

```
SQL>CREATE TABLE products (
prod_id NUMBER(3) CONSTRAINT p_ck CHECK (prod_id > 0),
prod_name CHAR(30),
prod_qty NUMBER(6),
CONSTRAINT p_name NOT NULL,
CONSTRAINT prod_pk PRIMARY KEY (prod_id));

SQL>CREATE TABLE warehouse (
warehouse_id NUMBER(4),
roomno NUMBER(10) CONSTRAINT r_id CHECK(roomno BETWEEN 101 AND 200),
location VARCHAR2(25),
prod_id NUMBER(3),
CONSTRAINT wr_pr_pk PRIMARY KEY (warehouse_id,prod_id),
CONSTRAINT prod_fk FOREIGN KEY (prod_id) REFERENCES products(prod_id));
```

Which statement is true regarding the execution of the above SQL commands?

A. Both commands execute successfully.
B. The first CREATE TABLE command generates an error because the NULL constraint is not valid.
C. The second CREATE TABLE command generates an error because the CHECK constraint is not valid.

D. The first CREATE TABLE command generates an error because CHECK and PRIMARY KEY constraints cannot be used for the same column.
E. The first CREATE TABLE command generates an error because the column PROD_ID cannot be used in the PRIMARY KEY and FOREIGN KEY constraints.

Answer: B
Section: (none)

Explanation/Reference:
Defining Constraints
The slide gives the syntax for defining constraints when creating a table. You can create constraints at either the column level or table level. Constraints defined at the column level are included when the column is defined. Table-level constraints are defined at the end of the table definition and must refer to the column or columns on which the constraint pertains in a set of parentheses. It is mainly the syntax that differentiates the two; otherwise, functionally, a columnlevel constraint is the same as a table-level constraint.
NOT NULL constraints must be defined at the column level.
Constraints that apply to more than one column must be defined at the table level

QUESTION 37
You issued the following command to drop the PRODUCTS table:

```
SQL> DROP TABLE products;
```

What is the implication of this command? (Choose all that apply.)

A. All data along with the table structure is deleted.
B. The pending transaction in the session is committed.
C. All indexes on the table will remain but they are invalidated.
D. All views and synonyms will remain but they are invalidated.
E. All data in the table are deleted but the table structure will remain.

Answer: ABD
Section: (none)

Explanation/Reference:

QUESTION 38
Which two statements are true regarding views? (Choose two.)

A. A simple view in which column aliases have been used cannot be updated.
B. Rows cannot be deleted through a view if the view definition contains the DISTINCT keyword.
C. Rows added through a view are deleted from the table automatically when the view is dropped.
D. The OR REPLACE option is used to change the definition of an existing view without dropping and re-creating it.
E. The WITH CHECK OPTION constraint can be used in a view definition to restrict the columns displayed through the view.

Answer: BD
Section: (none)

Explanation/Reference:

QUESTION 39
Evaluate the following command:

```
CREATE TABLE employees
(employee_id NUMBER(2) PRIMARY KEY,
last_name VARCHAR2(25) NOT NULL,
department_id NUMBER(2)NOT NULL,
job_id VARCHAR2(8),
salary NUMBER(10,2));
```

You issue the following command to create a view that displays the IDs and last names of the sales staff in the organization:

```
CREATE OR REPLACE VIEW sales_staff_vu
SELECT employee_id, last_name,job_id
FROM employees
WHERE job_id LIKE 'SA_%'
WITH CHECK OPTION;
```

Which two statements are true regarding the above view? (Choose two.)

A. It allows you to insert rows into the EMPLOYEES table .
B. It allows you to delete details of the existing sales staff from the EMPLOYEES table.
C. It allows you to update job IDs of the existing sales staff to any other job ID in the EMPLOYEES table.
D. It allows you to insert IDs, last names, and job IDs of the sales staff from the view if it is used in multitable INSERT statements.

Answer: BD
Section: (none)

Explanation/Reference:

QUESTION 40
View the Exhibit to examine the description for the SALES and PRODUCTS tables.

You want to create a SALE_PROD view by executing the following SQL statement:

```
CREATE VIEW sale_prod
AS SELECT p.prod_id, cust_id, SUM(quantity_sold) "Quantity" , SUM(prod_list_price) "Price"
FROM products p, sales s
WHERE p.prod_id=s.prod_id
GROUP BY p.prod_id, cust_id;
```

Which statement is true regarding the execution of the above statement?

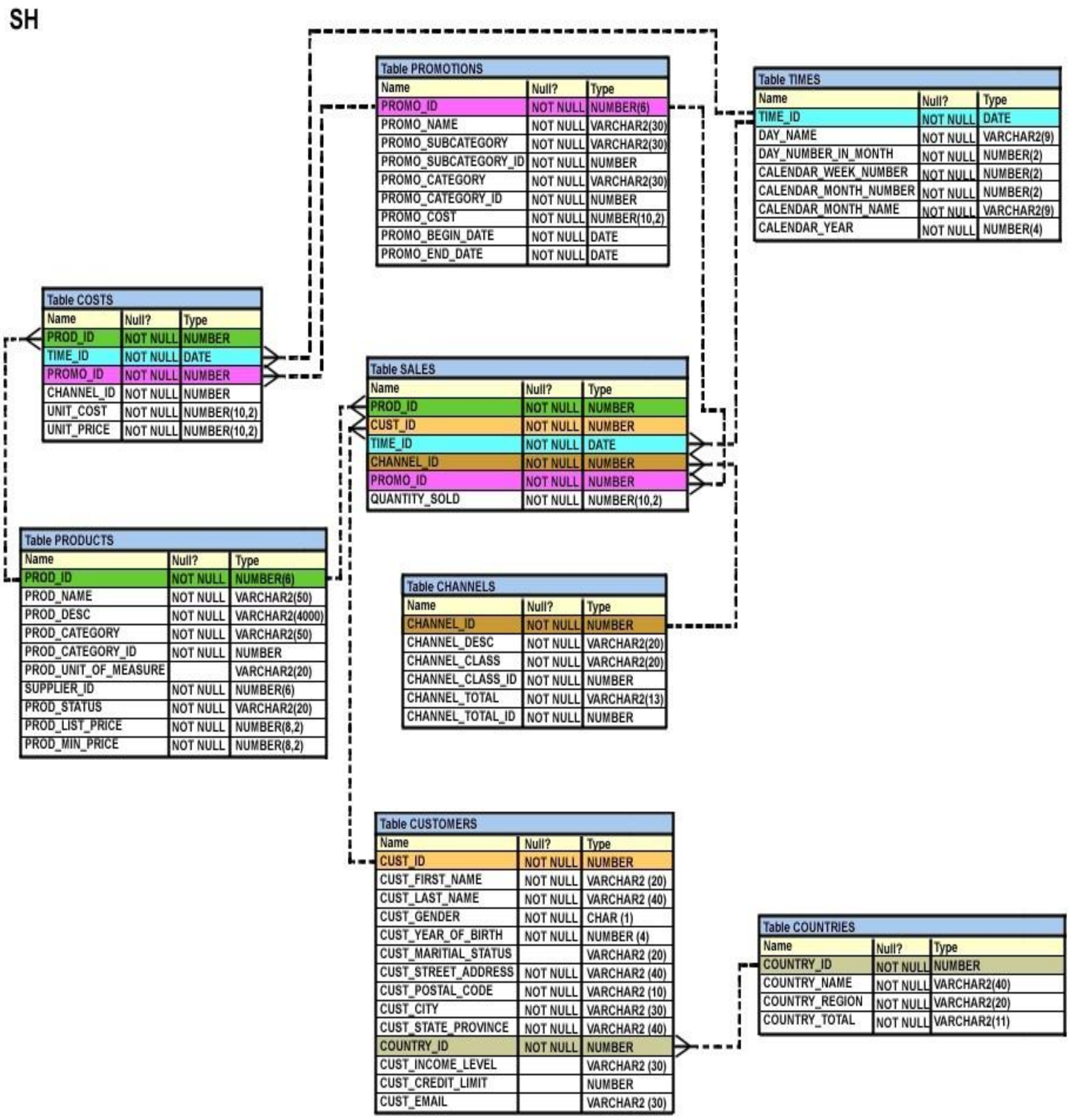


Table PROMOTIONS

Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

Table TIMES

Name	Null?	Type
TIME_ID	NOT NULL	DATE
DAY_NAME	NOT NULL	VARCHAR2(9)
DAY_NUMBER_IN_MONTH	NOT NULL	NUMBER(2)
CALENDAR_WEEK_NUMBER	NOT NULL	NUMBER(2)
CALENDAR_MONTH_NUMBER	NOT NULL	NUMBER(2)
CALENDAR_MONTH_NAME	NOT NULL	VARCHAR2(9)
CALENDAR_YEAR	NOT NULL	NUMBER(4)

Table COSTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
PROMO_ID	NOT NULL	NUMBER
CHANNEL_ID	NOT NULL	NUMBER
UNIT_COST	NOT NULL	NUMBER(10,2)
UNIT_PRICE	NOT NULL	NUMBER(10,2)

Table SALES

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
CUST_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
CHANNEL_ID	NOT NULL	NUMBER
PROMO_ID	NOT NULL	NUMBER
QUANTITY_SOLD	NOT NULL	NUMBER(10,2)

Table PRODUCTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

Table CHANNELS

Name	Null?	Type
CHANNEL_ID	NOT NULL	NUMBER
CHANNEL_DESC	NOT NULL	VARCHAR2(20)
CHANNEL_CLASS	NOT NULL	VARCHAR2(20)
CHANNEL_CLASS_ID	NOT NULL	NUMBER
CHANNEL_TOTAL	NOT NULL	VARCHAR2(13)
CHANNEL_TOTAL_ID	NOT NULL	NUMBER

Table CUSTOMERS

Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

Table COUNTRIES

Name	Null?	Type
COUNTRY_ID	NOT NULL	NUMBER
COUNTRY_NAME	NOT NULL	VARCHAR2(40)
COUNTRY_REGION	NOT NULL	VARCHAR2(20)
COUNTRY_TOTAL	NOT NULL	VARCHAR2(11)

A. The view will be created and you can perform DML operations on the view.
B. The view will be created but no DML operations will be allowed on the view.
C. The view will not be created because the join statements are not allowed for creating a view.
D. The view will not be created because the GROUP BY clause is not allowed for creating a view.

Answer: B
Section: (none)

Explanation/Reference:
Rules for Performing DML Operations on a View
You cannot add data through a view if the view includes:
  Group functions

A GROUP BY clause
The DISTINCT keyword
The pseudocolumn ROWNUM keyword
Columns defined by expressions
NOT NULL columns in the base tables that are not selected
by the view

QUESTION 41
Which two statements are true regarding views? (Choose two.)

A. A subquery that defines a view cannot include the GROUP BY clause.
B. A view that is created with the subquery having the DISTINCT keyword can be updated.
C. A view that is created with the subquery having the pseudo column ROWNUM keyword cannot be updated.
D. A data manipulation language (DML) operation can be performed on a view that is created with the subquery having all the NOT NULL columns of a table.

Answer: CD
Section: (none)

Explanation/Reference:
Rules for Performing DML Operations on a View
You cannot add data through a view if the view includes:
Group functions
A GROUP BY clause
The DISTINCT keyword
The pseudocolumn ROWNUM keyword
Columns defined by expressions
NOT NULL columns in the base tables that are not selected
by the view

QUESTION 42
Which three statements are true regarding views? (Choose three.)

A. Views can be created only from tables.
B. Views can be created from tables or other views.
C. Only simple views can use indexes existing on the underlying tables.
D. Both simple and complex views can use indexes existing on the underlying tables.
E. Complex views can be created only on multiple tables that exist in the same schema.
F. Complex views can be created on multiple tables that exist in the same or different schemas.

Answer: BDF
Section: (none)

Explanation/Reference:
Creating a Sequence (continued)
CYCLE | NOCYCLE Specifies whether the sequence continues to generate
values after reaching its maximum or minimum value
(NOCYCLE is the default option.)
CACHE **n** | NOCACHE Specifies how many values the Oracle server preallocates
and keeps in memory (By default, the Oracle server
caches 20 values.)

QUESTION 43
Evaluate the following CREATE SEQUENCE statement:

CREATE SEQUENCE seq1

```
START WITH 100
INCREMENT BY 10
MAXVALUE 200
CYCLE
NOCACHE;
```

The SEQ1 sequence has generated numbers up to the maximum limit of 200. You issue the following SQL statement:

SELECT seq1.nextval FROM dual;

What is displayed by the SELECT statement?

A. 1
B. 10
C. 100
D. an error

Answer: A
Section: (none)

Explanation/Reference:
But why the answer is not "C" ?
Because you didn't specify the MINVALUE for the sequence. If you check the sequence definition that you created it will have the default value of 1, which it reverts to when cycling.
If you wanted to keep the minimum value you would need to specify it in the sequence creation.

sequence Is the name of the sequence generator
INCREMENT BY **n** Specifies the interval between sequence numbers, where
n is an integer (If this clause is omitted, the sequence
increments by 1.)
START WITH **n** Specifies the first sequence number to be generated (If this
clause is omitted, the sequence starts with 1.)
MAXVALUE **n** Specifies the maximum value the sequence can generate
NOMAXVALUE Specifies a maximum value of 10^27 for an ascending
sequence and –1 for a descending sequence (This is the
default option.)
MINVALUE **n** Specifies the minimum sequence value
NOMINVALUE Specifies a minimum value of 1 for an ascending sequence
and –(10^26) for a descending sequence (This is the default
option.)
CYCLE **|** NOCYCLE Specifies whether the sequence continues to generate
values after reaching its maximum or minimum value
(NOCYCLE is the default option.)
CACHE **n** **|** NOCACHE Specifies how many values the Oracle server preallocates
and keeps in memory (By default, the Oracle server
caches 20 values.)

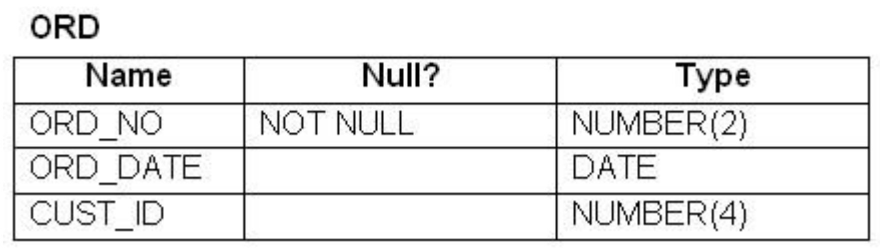
QUESTION 44
View the Exhibit and examine the structure of the ORD table.

Evaluate the following SQL statements that are executed in a user session in the specified order:

```
CREATE SEQUENCE ord_seq;
SELECT ord_seq.nextval
FROM dual;
INSERT INTO ord
VALUES (ord_seq.CURRVAL, '25-jan-2007',101);
UPDATE ord
```

```
SET ord_no= ord_seq.NEXTVAL
WHERE cust_id =101;
```

What would be the outcome of the above statements?



ORD

Name	Null?	Type
ORD_NO	NOT NULL	NUMBER(2)
ORD_DATE		DATE
CUST_ID		NUMBER(4)

A. All the statements would execute successfully and the ORD_NO column would contain the value 2 for the CUST_ID 101.
B. The CREATE SEQUENCE command would not execute because the minimum value and maximum value for the sequence have not been specified.
C. The CREATE SEQUENCE command would not execute because the starting value of the sequence and the increment value have not been specified.
D. All the statements would execute successfully and the ORD_NO column would have the value 20 for the CUST_ID 101 because the default CACHE value is 20.

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 45
Which two statements are true about sequences created in a single instance database? (Choose two.)

A. The numbers generated by a sequence can be used only for one table.
B. DELETE <sequencename> would remove a sequence from the database.
C. CURRVAL is used to refer to the last sequence number that has been generated.
D. When the MAXVALUE limit for a sequence is reached, you can increase the MAXVALUE limit by using the ALTER SEQUENCE statement.
E. When a database instance shuts down abnormally, the sequence numbers that have been cached but not used would be available once again when the database instance is restarted.

Answer: CD
Section: (none)

Explanation/Reference:

Gaps in the Sequence

Although sequence generators issue sequential numbers without gaps, this action occurs independent of a commit or rollback. Therefore, if you roll back a statement containing a sequence, the number is lost.
Another event that can cause gaps in the sequence is a system crash. If the sequence caches values in memory, those values are lost if the system crashes.
Because sequences are not tied directly to tables, the same sequence can be used for multiple tables. However, if you do so, each table can contain gaps in the sequential numbers.

Modifying a Sequence

If you reach the MAXVALUE limit for your sequence, no additional values from the sequence are allocated and you will receive an error indicating that the sequence exceeds the MAXVALUE. To continue to use the sequence, you can modify it by using the ALTER SEQUENCE statement

To remove a sequence, use the DROP statement:

DROP SEQUENCE dept_deptid_seq;

QUESTION 46
Which statements are correct regarding indexes? (Choose all that apply.)

A. When a table is dropped, the corresponding indexes are automatically dropped.
B. A FOREIGN KEY constraint on a column in a table automatically creates a nonunique index.
C. A nondeferrable PRIMARY KEY or UNIQUE KEY constraint in a table automatically creates a unique index.
D. For each data manipulation language (DML) operation performed, the corresponding indexes are automatically updated.

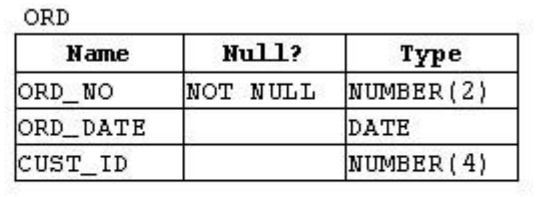
Answer: ACD
Section: (none)

Explanation/Reference:

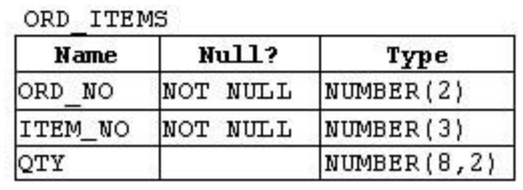
QUESTION 47
View the Exhibit and examine the structure of ORD and ORD_ITEMS tables.

The ORD_NO column is PRIMARY KEY in the ORD table and the ORD_NO and ITEM_NO columns are composite PRIMARY KEY in the ORD_ITEMS table.

Which two CREATE INDEX statements are valid? (Choose two.)


ORD

Name	Null?	Type
ORD_NO	NOT NULL	NUMBER(2)
ORD_DATE		DATE
CUST_ID		NUMBER(4)


ORD_ITEMS

Name	Null?	Type
ORD_NO	NOT NULL	NUMBER(2)
ITEM_NO	NOT NULL	NUMBER(3)
QTY		NUMBER(8,2)

A. CREATE INDEX ord_idx1
ON ord(ord_no);
B. CREATE INDEX ord_idx2
ON ord_items(ord_no);
C. CREATE INDEX ord_idx3
ON ord_items(item_no);
D. CREATE INDEX ord_idx4
ON ord,ord_items(ord_no, ord_date,qty);

Answer: BC
Section: (none)

Explanation/Reference:
How Are Indexes Created?
You can create two types of indexes.
Unique index: The Oracle server automatically creates this index when you define a column in a table to have a PRIMARY KEY or a UNIQUE constraint. The name of the index is the name that is given to the constraint.
Nonunique index: This is an index that a user can create. For example, you can create the FOREIGN KEY column index for a join in a query to improve the speed of retrieval.

Note: You can manually create a unique index, but it is recommended that you create a unique constraint, which implicitly creates a unique index.

QUESTION 48
Which two statements are true regarding indexes? (Choose two.)

A. They can be created on tables and clusters.
B. They can be created on tables and simple views.
C. You can create only one index by using the same columns.
D. You can create more than one index by using the same columns if you specify distinctly different combinations of the columns.

Answer: AD
Section: (none)

Explanation/Reference:

QUESTION 49
The ORDERS table belongs to the user OE. OE has granted the SELECT privilege on the ORDERS table to the user HR.

Which statement would create a synonym ORD so that HR can execute the following query successfully?

SELECT * FROM ord;

A. CREATE SYNONYM ord FOR orders; This command is issued by OE.
B. CREATE PUBLIC SYNONYM ord FOR orders; This command is issued by OE.
C. CREATE SYNONYM ord FOR oe.orders; This command is issued by the database administrator.
D. CREATE PUBLIC SYNONYM ord FOR oe.orders; This command is issued by the database administrator.

Answer: D
Section: (none)

Explanation/Reference:
Creating a Synonym for an Object
To refer to a table that is owned by another user, you need to prefix the table name with the name of the user who created it, followed by a period. Creating a synonym eliminates the need to qualify the object name with the schema and provides you with an alternative name for a table, view, sequence, procedure, or other objects. This method can be especially useful with lengthy object names, such as views.
In the syntax:
PUBLIC Creates a synonym that is accessible to all users
synonym Is the name of the synonym to be created
object Identifies the object for which the synonym is created
Guidelines
The object cannot be contained in a package.
A private synonym name must be distinct from all other objects that are owned by the same user.

If you try to execute the following command (alternative B, issued by OE):

CREATE PUBLIC SYNONYM ord FOR orders;

You will get an error.

Error que empieza en la línea 693 del comando:
create public synonym nuly for prueba_null
Error en la línea de comandos:693 Columna:0

Informe de error:
Error SQL: ORA-01031: privilegios insuficientes
01031. 00000 - "insufficient privileges"

The message gives you the answer: OE doesn't have enough privileges. However, if you give the necessary privileges (issued by DBA):

GRANT **CREATE PUBLIC SYNONYM** TO **OE**;

You won't have problems executing the command in the alternative B (issued by OE): CREATE PUBLIC SYNONYM ord FOR orders;

Finally, if you need to be sure what system privileges you have in your active session, you can execute the following command (issued by OE):

SELECT * FROM USER_PRIVS;

(One of the rows must be: **CREATE PUBLIC SYNONYM**).

QUESTION 50
SLS is a private synonym for the SH.SALES table.

The user SH issues the following command:

DROP SYNONYM sls;

Which statement is true regarding the above SQL statement?

A. Only the synonym would be dropped.
B. The synonym would be dropped and the corresponding table would become invalid.
C. The synonym would be dropped and the packages referring to the synonym would be dropped.
D. The synonym would be dropped and any PUBLIC synonym with the same name becomes invalid.

Answer: A
Section: (none)

Explanation/Reference:
A synonym is an alias for
a table (or a view). Users can execute SQL statements against the synonym, and
the database will map them into statements against the object to which the synonym
points.
Private synonyms are schema objects. Either they must be in your own schema, or they must be qualified with the schema name. Public synonyms exist independently of a schema. A public synonym can be referred to by any user to whom permission has been granted to see it without the need to qualify it with a schema name. Private synonyms must be a unique name within their schema**. Public synonyms can have the same name as schema objects**. When executing statements that address objects without a schema qualifier, Oracle will first look for the object in the local schema, and only if it cannot be found will it look for a public synonym.

QUESTION 51
Which statement is true regarding synonyms?

A. Synonyms can be created only for a table.
B. Synonyms are used to reference only those tables that are owned by another user.
C. A public synonym and a private synonym can exist with the same name for the same table.
D. The DROP SYNONYM statement removes the synonym, and the table on which the synonym has been created becomes invalid.

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 52
View the Exhibit and examine the structure of the PRODUCTS table.

Using the PRODUCTS table, you issue the following query to generate the names, current list price, and discounted list price for all those products whose list price falls below $10 after a discount of 25% is applied on it.

```
SQL>SELECT prod_name, prod_list_price,
prod_list_price - (prod_list_price * .25) "DISCOUNTED_PRICE"
FROM products
WHERE discounted_price < 10;
```

The query generates an error.
What is the reason for the error?

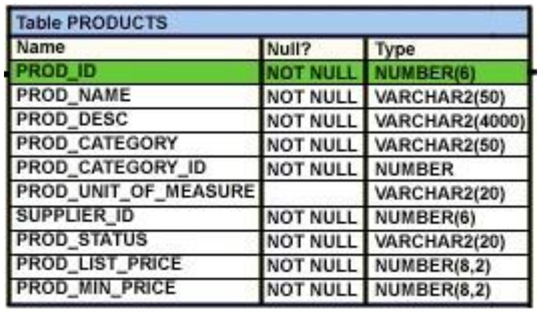


Table PRODUCTS		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

A. The parenthesis should be added to enclose the entire expression.
B. The double quotation marks should be removed from the column alias.
C. The column alias should be replaced with the expression in the WHERE clause.
D. The column alias should be put in uppercase and enclosed with in double quotation marks in the WHERE clause.

Answer: C
Section: (none)

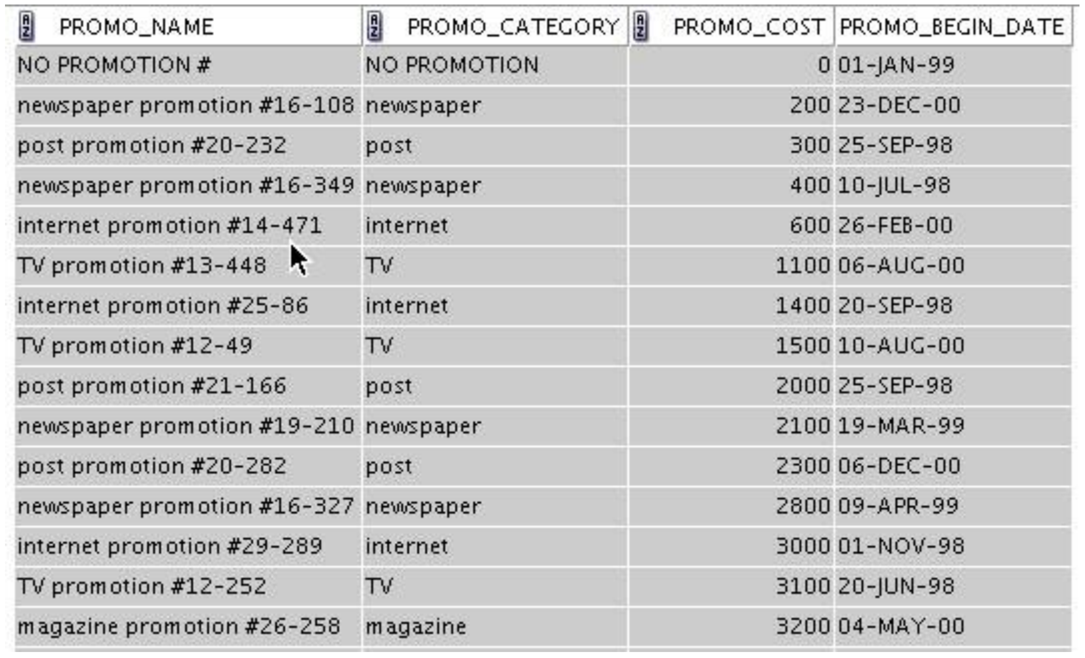
Explanation/Reference:
Note: You cannot use column alias in the WHERE clause.

QUESTION 53
View the Exhibit and examine the data in the PROMOTIONS table.

PROMO_BEGIN_DATE is stored in the default date format, **dd-mon-rr**.

You need to produce a report that provides the name, cost, and start date of all promos in the POST category that were launched before January 1, 2000.

Which SQL statement would you use?



PROMO_NAME	PROMO_CATEGORY	PROMO_COST	PROMO_BEGIN_DATE
NO PROMOTION #	NO PROMOTION	0	01-JAN-99
newspaper promotion #16-108	newspaper	200	23-DEC-00
post promotion #20-232	post	300	25-SEP-98
newspaper promotion #16-349	newspaper	400	10-JUL-98
internet promotion #14-471	internet	600	26-FEB-00
TV promotion #13-448	TV	1100	06-AUG-00
internet promotion #25-86	internet	1400	20-SEP-98
TV promotion #12-49	TV	1500	10-AUG-00
post promotion #21-166	post	2000	25-SEP-98
newspaper promotion #19-210	newspaper	2100	19-MAR-99
post promotion #20-282	post	2300	06-DEC-00
newspaper promotion #16-327	newspaper	2800	09-APR-99
internet promotion #29-289	internet	3000	01-NOV-98
TV promotion #12-252	TV	3100	20-JUN-98
magazine promotion #26-258	magazine	3200	04-MAY-00

A. SELECT promo_name, promo_cost, promo_begin_date
   FROM promotions
   WHERE promo_category = 'post' AND promo_begin_date < '01-01-00';
B. SELECT promo_name, promo_cost, promo_begin_date
   FROM promotions
   WHERE promo_cost LIKE 'post%' AND promo_begin_date < '01-01-2000';
C. SELECT promo_name, promo_cost, promo_begin_date
   FROM promotions
   WHERE promo_category LIKE 'P%' AND promo_begin_date < '1-JANUARY-00';
D. SELECT promo_name, promo_cost, promo_begin_date
   FROM promotions
   WHERE promo_category LIKE '%post%' AND promo_begin_date < '1-JAN-00';

Answer: D
Section: (none)

Explanation/Reference:

QUESTION 54
View the Exhibit and examine the structure of the CUSTOMERS table.

Evaluate the query statement:

```
SQL> SELECT cust_last_name, cust_city, cust_credit_limit
FROM customers
WHERE cust_last_name BETWEEN 'A' AND 'C' AND cust_credit_limit BETWEEN
1000 AND 3000;
```

What would be the outcome of the above statement?

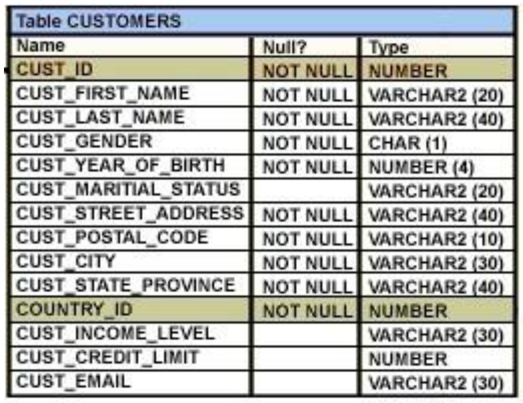


Table CUSTOMERS		
Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

A. It executes successfully.
B. It produces an error because the condition on CUST_LAST_NAME is invalid.
C. It executes successfully only if the CUST_CREDIT_LIMIT column does not contain any null values.
D. It produces an error because the AND operator cannot be used to combine multiple BETWEEN clauses.

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 55
Evaluate the following two queries:

```
SQL> SELECT cust_last_name, cust_city
FROM customers
WHERE cust_credit_limit IN (1000, 2000, 3000);

SQL> SELECT cust_last_name, cust_city
FROM customers
WHERE cust_credit_limit = 1000 OR cust_credit_limit = 2000 OR
cust_credit_limit = 3000;
```

Which statement is true regarding the above two queries?

A. Performance would improve in query 2.
B. Performance would degrade in query 2.
C. There would be no change in performance.
D. Performance would improve in query 2 only if there are null values in the CUST_CREDIT_LIMIT column.

Answer: C
Section: (none)

Explanation/Reference:
Note: The IN operator is internally evaluated by the Oracle server as a set of OR conditions, such as a=value1 or a=value2 or a=value3. Therefore, using the IN operator has no performance benefits and is used only for logical simplicity.

QUESTION 56
View the Exhibit and examine the structure of the PROMOTIONS table.

Using the PROMOTIONS table, you need to find out the names and cost of all the promos done on 'TV' and 'internet' that ended in the time interval 15th March '00 to 15th October '00.

Which two queries would give the required result? (Choose two.)

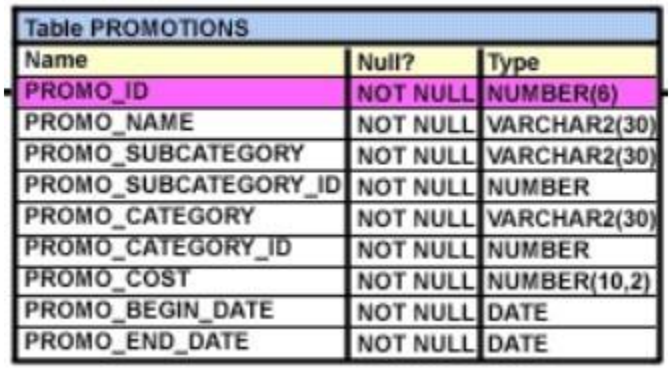


Table PROMOTIONS		
Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

A. SELECT promo_name, promo_cost
FROM promotions
WHERE promo_category IN ('TV', 'internet') AND
promo_end_date BETWEEN '15-MAR-00' AND '15-OCT-00';

B. SELECT promo_name, promo_cost
FROM promotions
WHERE promo_category = 'TV' OR promo_category ='internet' AND promo_end_date >='15-MAR-00' OR
promo_end_date <='15-OCT-00';

C. SELECT promo_name, promo_cost
FROM promotions
WHERE (promo_category BETWEEN 'TV' AND 'internet') AND
(promo_end_date IN ('15-MAR-00','15-OCT-00'));

D. SELECT promo_name, promo_cost
FROM promotions
WHERE (promo_category = 'TV' OR promo_category ='internet') AND (promo_end_date >='15-MAR-00' AND
promo_end_date <='15-OCT-00');

Answer: AD
Section: (none)

Explanation/Reference:

QUESTION 57
The CUSTOMERS table has the following structure:

```
Name                                Null          Type
----------------------------------- ------------- -------------
CUST_ID                             NOT NULL      NUMBER
CUST_FIRST_NAME                     NOT NULL      VARCHAR2(20)
CUST_LAST_NAME                      NOT NULL      VARCHAR2(30)
CUST_INCOME_LEVEL                                 VARCHAR2(30)
CUST_CREDIT_LIMIT                                 NUMBER
```

You need to write a query that does the following tasks:
1. Display the first name and tax amount of the customers. Tax is 5% of their credit limit.
2. Only those customers whose income level has a value should be considered.
3. Customers whose tax amount is null should not be considered.
Which statement accomplishes all the required tasks?

Which statement accomplishes all the required tasks?

A. SELECT cust_first_name, cust_credit_limit * .05 AS TAX_AMOUNT FROM customers
WHERE cust_income_level IS NOT NULL AND
tax_amount IS NOT NULL;

B. SELECT cust_first_name, cust_credit_limit * .05 AS TAX_AMOUNT FROM customers

WHERE cust_income_level IS NOT NULL AND
   cust_credit_limit IS NOT NULL;
C. SELECT cust_first_name, cust_credit_limit * .05 AS TAX_AMOUNT FROM customers
   WHERE cust_income_level <> NULL AND
   tax_amount <> NULL;
D. SELECT cust_first_name, cust_credit_limit * .05 AS TAX_AMOUNT FROM customers
   WHERE (cust_income_level,tax_amount) IS NOT NULL;

Answer: B
Section: (none)

Explanation/Reference:

QUESTION 58
The PART_CODE column in the SPARES table contains the following list of values:

```
PART_CODE
---------------------
A%_WQ123
A%BWQ123
AB_WQ123
```

Evaluate the following query:

```
SQL> SELECT part_code
FROM spares
WHERE part_code LIKE '%\%_WQ12%' ESCAPE '\';
```

Which statement is true regarding the outcome of the above query?

A. It produces an error.
B. It displays all values.
C. It displays only the values A%_WQ123 and AB_WQ123 .
D. It displays only the values A%_WQ123 and A%BWQ123 .
E. It displays only the values A%BWQ123 and AB_WQ123.

Answer: D
Section: (none)

Explanation/Reference:
Combining Wildcard Characters
The % and _ symbols can be used in any combination with literal characters. The example in the slide displays the names of all employees whose last names have the letter “o” as the second character.
ESCAPE Identifier
When you need to have an exact match for the actual **%** and _ characters, use the ESCAPE identifier. This option specifies what the escape character is. If you want to search for strings that contain SA_, you can use the following SQL statement:
SELECT employee_id, last_name, job_id
FROM employees WHERE job_id LIKE '%SA_%' ESCAPE '\';

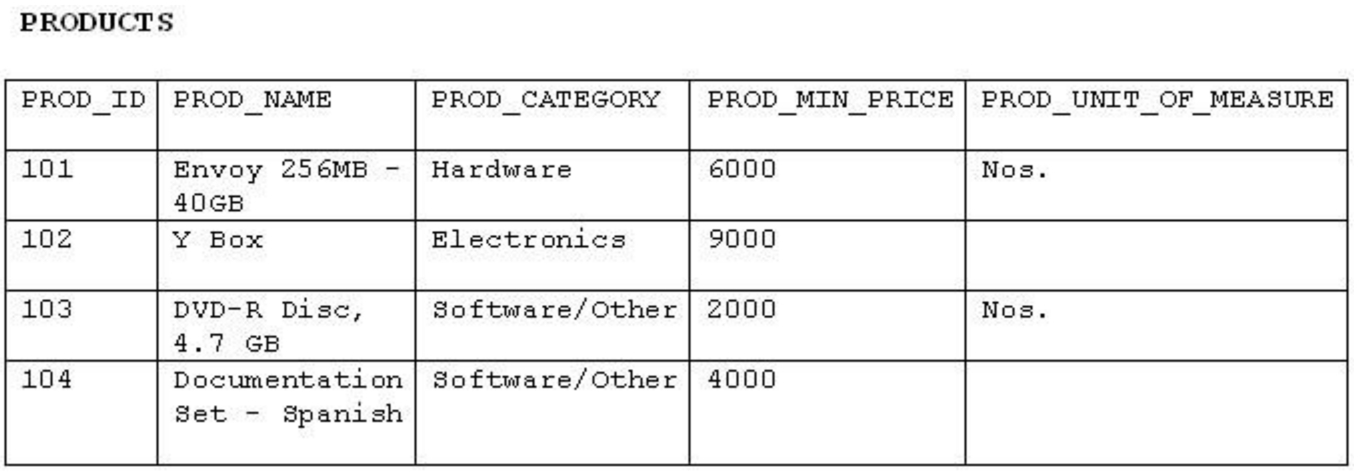
QUESTION 59
View the Exhibit and examine the data in the PRODUCTS table.

You need to display product names from the PRODUCTS table that belong to the 'Software/Other ' category with minimum prices as either $2000 or $4000 and no unit of measure.

You issue the following query:

```
SQL>SELECT prod_name, prod_category, prod_min_price
FROM products
WHERE prod_category LIKE '%Other%' AND (prod_min_price = 2000 OR
prod_min_price = 4000) AND prod_unit_of_measure <> '';
```

Which statement is true regarding the above query?



PRODUCTS

PROD_ID	PROD_NAME	PROD_CATEGORY	PROD_MIN_PRICE	PROD_UNIT_OF_MEASURE
101	Envoy 256MB - 40GB	Hardware	6000	Nos.
102	Y Box	Electronics	9000	
103	DVD-R Disc, 4.7 GB	Software/Other	2000	Nos.
104	Documentation Set - Spanish	Software/Other	4000	

A. It executes successfully but returns no result.
B. It executes successfully and returns the required result.
C. It generates an error because the condition specified for PROD_UNIT_OF_MEASURE is not valid.
D. It generates an error because the condition specified for the PROD_CATEGORY column is not valid.

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 60
View the Exhibit and examine the structure of CUSTOMERS table.

Evaluate the following query:

```
SQL>SELECT cust_id, cust_city
FROM customers
WHERE cust_first_name NOT LIKE 'A_%g_%' AND
cust_credit_limit BETWEEN 5000 AND 15000 AND
cust_credit_limit NOT IN (7000, 11000) AND
cust_city NOT BETWEEN 'A' AND 'B';
```

Which statement is true regarding the above query?

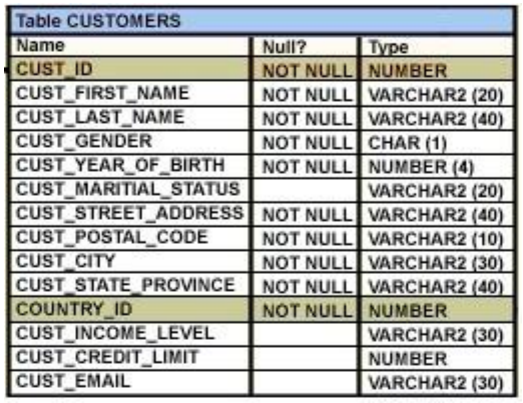


Table CUSTOMERS		
Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

A. It executes successfully.
B. It produces an error because the condition on the CUST_CITY column is not valid.
C. It produces an error because the condition on the CUST_FIRST_NAME column is not valid.
D. It produces an error because conditions on the CUST_CREDIT_LIMIT column are not valid.

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 61
View the Exhibit and examine the structure of the PROMOTIONS table.
You need to generate a report of all promos from the PROMOTIONS table based on the following conditions:
1. The promo name should not begin with 'T' or 'N'.
2. The promo should cost more than $20000.
3. The promo should have ended after 1st January 2001.

Which WHERE clause would give the required result?

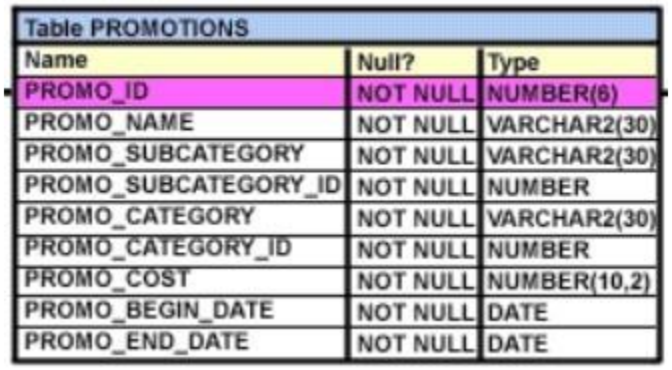


Table PROMOTIONS		
Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

A. WHERE promo_name NOT LIKE 'T%' OR promo_name NOT LIKE 'N%' AND promo_cost > 20000 AND promo_end_date > '1-JAN-01'
B. WHERE (promo_name NOT LIKE 'T%' AND promo_name NOT LIKE 'N%')OR promo_cost > 20000 OR promo_end_date > '1-JAN-01'
C. WHERE promo_name NOT LIKE 'T%' AND promo_name NOT LIKE 'N%' AND promo_cost > 20000 AND promo_end_date > '1-JAN-01'
D. WHERE (promo_name NOT LIKE '%T%' OR promo_name NOT LIKE '%N%') AND(promo_cost > 20000 AND promo_end_date > '1-JAN-01')

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 62
View the E xhibit and examine the structure of the CUSTOMERS table.

You want to generate a report showing the last names and credit limits of all customers whose last names start with A, B, or C, and credit limit is below 10, 000.

Evaluate the following two queries:

```
SQL> SELECT cust_last_name, cust_credit_limit FROM customers
WHERE (UPPER(cust_last_name) LIKE 'A%' OR
UPPER(cust_last_name) LIKE 'B%' OR UPPER(cust_last_name) LIKE 'C%')
AND cust_credit_limit < 10000;

SQL>SELECT cust_last_name, cust_credit_limit FROM customers
WHERE UPPER(cust_last_name) BETWEEN 'A' AND 'C'
AND cust_credit_limit < 10000;
```

Which statement is true regarding the execution of the above queries?

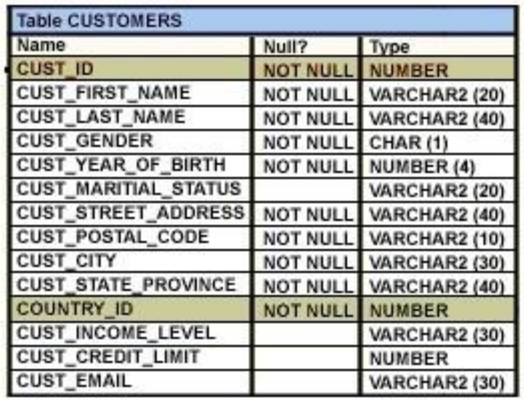


Table CUSTOMERS		
Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

A. Only the first query gives the correct result.
B. Only the second query gives the correct result.
C. Both execute successfully and give the same result.
D. Both execute successfully but do not give the required result.

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 63
View the Exhibit and examine the structure of the PRODUCTS table.

You want to display only those product names with their list prices where the list price is at least double the minimum price.
The report should start with the product name having the maximum list price satisfying this condition.

Evaluate the following SQL statement:

```
SQL>SELECT prod_name,prod_list_price
FROM products
```

WHERE prod_list_price >= 2 * prod_min_price
Which ORDER BY clauses can be added to the above SQL statement to get the correct output?

(Choose all that apply.)

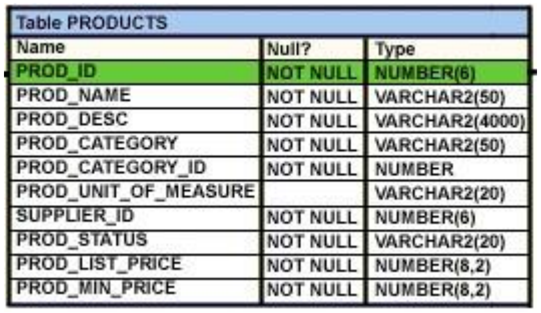


Table PRODUCTS		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

A. ORDER BY prod_list_price DESC, prod_name;
B. ORDER BY (2*prod_min_price)DESC, prod_name;
C. ORDER BY prod_name, (2*prod_min_price)DESC;
D. ORDER BY prod_name DESC, prod_list_price DESC;
E. ORDER BY prod_list_price DESC, prod_name DESC;

Answer: AE
Section: (none)

Explanation/Reference:
Using the ORDER BY Clause
The order of rows that are returned in a query result is undefined. The ORDER BY clause can be used to sort the rows. However, if you use the ORDER BY clause, it must be the last clause of the SQL statement. Further, you can specify an expression, an alias, or a column position as the sort condition.
Syntax
SELECT **expr**
FROM **table**
[WHERE **condition(s)**]
[ORDER BY {**column**, **expr, numeric_position**} [ASC|DESC]];
In the syntax:
ORDER BY specifies the order in which the retrieved rows are displayed
ASC orders the rows in ascending order (This is the default order.)
DESC orders the rows in descending order
If the ORDER BY clause is not used, the sort order is undefined, and the Oracle server may not fetch rows in the same order for the same query twice. Use the ORDER BY clause to display the rows in a specific order.
Note: Use the keywords NULLS FIRST or NULLS LAST to specify whether returned rows containing null values should appear first or last in the ordering sequence.

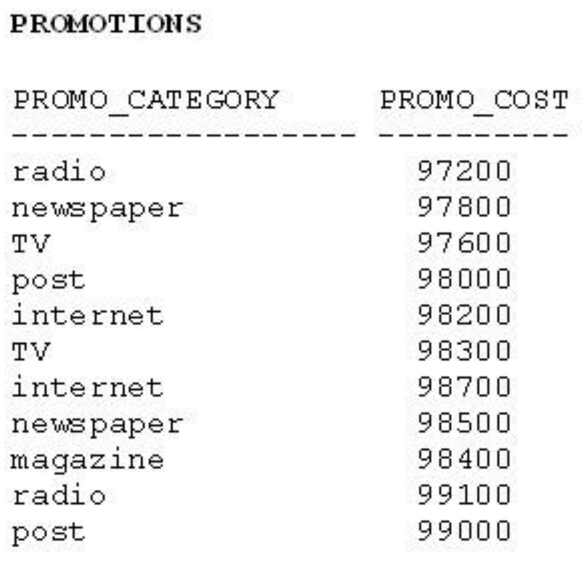
QUESTION 64
View the Exhibit and examine the data in the PROMO_CATEGORY and PROMO_COST columns of the PROMOTIONS table.

Evaluate the following two queries:

```
SQL>SELECT DISTINCT promo_category to_char(promo_cost)"code"
FROM promotions
ORDER BY code;
```

```
SQL>SELECT DISTINCT promo_category promo_cost "code"
FROM promotions
ORDER BY 1;
```

Which statement is true regarding the execution of the above queries?



```
PROMOTIONS

PROMO_CATEGORY     PROMO_COST
------------------ ----------
radio                  97200
newspaper              97800
TV                     97600
post                   98000
internet               98200
TV                     98300
internet               98700
newspaper              98500
magazine               98400
radio                  99100
post                   99000
```

A. Only the first query executes successfully.
B. Only the second query executes successfully.
C. Both queries execute successfully but give different results.
D. Both queries execute successfully and give the same result.

Answer: B
Section: (none)

Explanation/Reference:
Note: You cannot use column alias in the WHERE clause.

QUESTION 65
View the Exhibit and examine the structure of the CUSTOMERS table.

You have been asked to produce a report on the CUSTOMERS table showing the customers details sorted in descending order of the city and in the descending order of their income level in each city.

Which query would accomplish this task?

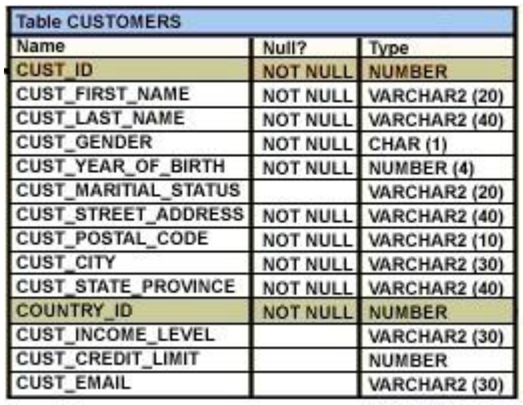


Table CUSTOMERS

Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

A. SELECT cust_city, cust_income_level, cust_last_name

```
FROM customers
ORDER BY cust_city desc, cust_income_level DESC ;
```

B.
```
SELECT cust_city, cust_income_level, cust_last_name
FROM customers
ORDER BY cust_income_level desc, cust_city DESC;
```

C.
```
SELECT cust_city, cust_income_level, cust_last_name
FROM customers
ORDER BY (cust_city, cust_income_level) DESC;
```

D.
```
SELECT cust_city, cust_income_level, cust_last_name
FROM customers
ORDER BY cust_city, cust_income_level DESC;
```

Answer: A
Section: (none)

Explanation/Reference:

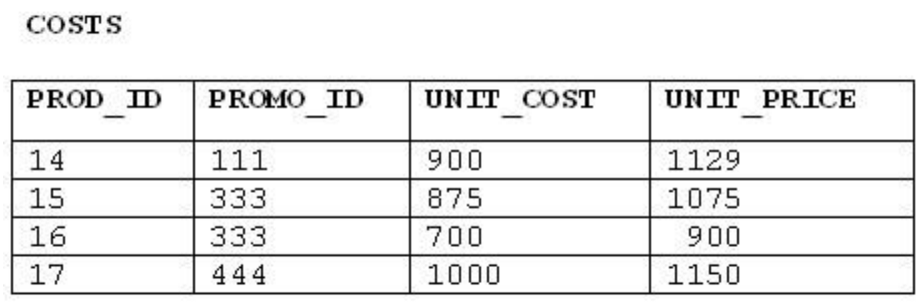
QUESTION 66
View the Exhibit and examine the data in the COSTS table.

You need to generate a report that displays the IDs of all products in the COSTS table whose unit price is at least 25% more than the unit cost. The details should be displayed in the descending order of 25% of the unit cost.

You issue the following query:

```
SQL>SELECT prod_id
FROM costs
WHERE unit_price >= unit_cost * 1.25
ORDER BY unit_cost * 0.25 DESC;
```

Which statement is true regarding the above query?


COSTS

PROD_ID	PROMO_ID	UNIT_COST	UNIT_PRICE
14	111	900	1129
15	333	875	1075
16	333	700	900
17	444	1000	1150

A. It executes and produces the required result.
B. It produces an error because an expression cannot be used in the ORDER BY clause.
C. It produces an error because the DESC option cannot be used with an expression in the ORDER BY clause.
D. It produces an error because the expression in the ORDER BY clause should also be specified in the SELECT clause.

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 67
Which two statements are true regarding the ORDER BY clause? (Choose two.)

A. It is executed first in the query execution.

B. It must be the last clause in the SELECT statement.
C. It cannot be used in a SELECT statement containin g a HAVING clause.
D. You cannot specify a column name followed by an expression in this clause.
E. You can specify a combination of numeric positions and column names in this clause.

Answer: BE
Section: (none)

Explanation/Reference:

QUESTION 68
Which statement is true regarding the default behavior of the ORDER BY clause?

A. In a character sort, the values are case- sensitive.
B. NULL values are not considered at all by the sort operation.
C. Only those columns that are specified in the SELECT list can be used in the ORDER BY clause.
D. Numeric values are displayed from the maximum to the minimum value if they have decimal positions.

Answer: A
Section: (none)

Explanation/Reference:
Character Strings and Dates
Character strings and date values are enclosed with single quotation marks.
Character values **are case-sensitive** and date values are format-sensitive.
The default date display format is DD-MON-RR.

QUESTION 69
You need to generate a list of all customer last names with their credit limits from the CUSTOMERS table.
Those customers who do not have a credit limit should appear last in the list.

Which two queries would achieve the required result? (Choose two.)

A. SELECT cust_last_name, cust_credit_limit
FROM customers
ORDER BY cust_credit_limit DESC ;
B. SELECT cust_last_name, cust_credit_limit
FROM customers
ORDER BY cust_credit_limit;
C. SELECT cust_last_name, cust_credit_limit
FROM customers
ORDER BY cust_credit_limit NULLS LAST;
D. SELECT cust_last_name, cust_credit_limit
FROM customers
ORDER BY cust_last_name, cust_credit_limit NULLS LAST;

Answer: BC
Section: (none)

Explanation/Reference:
If the ORDER BY clause is not used, the sort order is undefined, and the Oracle server may not fetch rows in the same order for the same query twice. Use the ORDER BY clause to display the rows in a specific order.
Note: Use the keywords NULLS FIRST or NULLS LAST to specify whether returned rows containing null values should appear first or last in the ordering sequence. ANSWER C
Sorting
The default sort order is ascending:

• Numeric values are displayed with the lowest values first (for example, 1 to 999).
• Date values are displayed with the earliest value first (for example, 01-JAN-92 before 01-JAN-95).
• Character values are displayed in the alphabetical order (for example, “A” first and “Z” last).
• Null values are displayed last for ascending sequences and first for descending sequences. - ANSWER B
• You can also sort by a column that is not in the SELECT list.

QUESTION 70
View the Exhibit and examine the structure of the PRODUCTS table.

You want to display only those product names with their list prices where the list price is at least double the minimum price.
The report should start with the product name having the maximum list price satisfying this condition.

Evaluate the following SQL statement:

SQL>SELECT prod_name,prod_list_price
FROM products
WHERE prod_list_price >= 2 * prod_min_price
Which ORDER BY clauses can be added to the above SQL statement to get the correct output?

(Choose all that apply.)

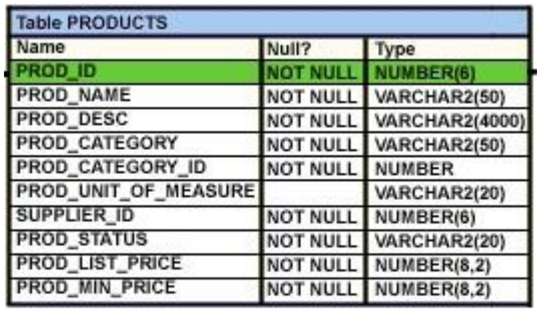


Table PRODUCTS		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

A. ORDER BY prod_list_price DESC, prod_name;
B. ORDER BY (2*prod_min_price)DESC, prod_name;
C. ORDER BY prod_name, (2*prod_min_price)DESC;
D. ORDER BY prod_name DESC, prod_list_price DESC;
E. ORDER BY prod_list_price DESC, prod_name DESC;

Answer: AE
Section: (none)

Explanation/Reference:

QUESTION 71
Which arithmetic operations can be performed on a column by using a SQL function that is built into Oracle database ? (Choose three .)

A. addition
B. subtraction
C. raising to a power
D. finding the quotient
E. finding the lowest value

Answer: ACE

Section: (none)

Explanation/Reference:

QUESTION 72
Which tasks can be performed using SQL functions built into Oracle Database ? (Choose three.)

A. displaying a date in a nondefault format
B. finding the number of characters in an expression
C. substituting a character string in a text expression with a specified string
D. combining more than two columns or expressions into a single column in the output

Answer: ABC
Section: (none)

Explanation/Reference:

QUESTION 73
Which tasks can be performed using SQL functions that are built into Oracle database ? (Choose three .)

A. finding the remainder of a division
B. adding a number to a date for a resultant date value
C. comparing two expressions to check whether they are equal
D. checking whether a specified character exists in a given string
E. removing trailing, leading, and embedded characters from a character string

Answer: ACD
Section: (none)

Explanation/Reference:

QUESTION 74
Which statements are true regarding single row functions? (Choose all that apply.)

A. MOD : returns the quotient of a division
B. TRUNC : can be used with NUMBER and DATE values
C. CONCAT : can be used to combine any number of values
D. SYSDATE : returns the database server current date and time
E. INSTR : can be used to find only the first occurrence of a character in a string
F. TRIM : can be used to remove all the occurrences of a character from a string

Answer: BD
Section: (none)

Explanation/Reference:
ROUND: Rounds value to a specified decimal
TRUNC: Truncates value to a specified decimal
MOD: Returns remainder of division

SYSDATE is a date function that returns the current database server date and time.

Date-Manipulation Functions

Date functions operate on Oracle dates. All date functions return a value of the DATE data type except MONTHS_BETWEEN, which returns a numeric value.

MONTHS_BETWEEN(**date1, date2**): Finds the number of months between **date1** and **date2**. The result can be

positive or negative. If **date1** is later than **date2**, the result is positive; if **date1** is earlier than **date2**, the result is negative. The noninteger part of the result represents a portion of the month.
ADD_MONTHS(**date, n**): Adds **n** number of calendar months to **date**. The value of **n** must be an integer and can be negative.
NEXT_DAY(**date,** '**char**'): Finds the date of the next specified day of the week ('**char**') following **date**. The value of **char** may be a number representing a day or a character string.
LAST_DAY(**date**): Finds the date of the last day of the month that contains **date**
The above list is a subset of the available date functions. ROUND and TRUNC number functions can also be used to manipulate the date values as shown below:
ROUND(**date**[,'**fmt**']): Returns **date** rounded to the unit that is specified by the format model **fmt.** If the format model **fmt** is omitted, **date** is rounded to the nearest day.
_TRUNC(date[, 'fmt']): Returns **date** with the time portion of the day truncated to the unit that is specified by the format model **fmt**. If the format model **fmt** is omitted, **date** is truncated to the nearest day.

The CONCAT Function

The CONCAT function joins **two** character literals, columns, or expressions to yield one larger character expression. Numeric and date literals are implicitly cast as characters when they occur as parameters to the CONCAT function. Numeric or date expressions are evaluated before being converted to strings ready to be concatenated. The CONCAT function takes two parameters. Its syntax is CONCAT(s1, s2), where s1 and s2 represent string literals, character column values, or expressions resulting in character values.
The **INSTR**(source string, search item, [start position],[**nth** occurrence of search item]) function returns a number that represents the position in the source string, beginning from the given start position, where the **nth** occurrence of the search item begins:
instr('http://www.domain.com','.',1,2) = 18
The **TRIM** function literally trims off leading or trailing (or both) character strings from a given source string:

QUESTION 75
The following data exists in the PRODUCTS table:

```
PROD_ID PROD_LIST_PRICE
---------------------------------------------
123456 152525.99
```

You issue the following query:

```
SQL> SELECT RPAD(( ROUND(prod_list_price)), 10,'*')
FROM products
WHERE prod_id = 123456;
```

What would be the outcome?

A. 152526 ****
B. **152525.99
C. 152525** **
D. an error message

Answer: A
Section: (none)

Explanation/Reference:
The **LPAD**(string, length after padding, padding string) and **RPAD**(string, length after padding, padding string) functions add a padding string of characters to the left or right of a string until it reaches the specified length after padding.

QUESTION 76
You need to display the first names of all customers from the CUSTOMERS table that contain the character 'e' and have the character 'a' in the second last position.

Which query would give the required output?

A. SELECT cust_first_name
   FROM customers
   WHERE INSTR(cust_first_name, 'e')<>0 AND
   SUBSTR(cust_first_name, -2, 1)='a';
B. SELECT cust_first_name
   FROM customers
   WHERE INSTR(cust_first_name, 'e')<>'' AND
   SUBSTR(cust_first_name, -2, 1)='a';
C. SELECT cust_first_name
   FROM customers
   WHERE INSTR(cust_first_name, 'e')IS NOT NULL AND
   SUBSTR(cust_first_name, 1,-2)='a';
D. SELECT cust_first_name
   FROM customers
   WHERE INSTR(cust_first_name, 'e')<>0 AND
   SUBSTR(cust_first_name, LENGTH(cust_first_name),-2)='a';

Answer: A
Section: (none)

Explanation/Reference:
The **SUBSTR**(string, start position, number of characters) function accepts three parameters and returns a string consisting of the number of characters extracted from the source string, beginning at the specified start position:
substr('http://www.domain.com',12,6) = domain
The position at which the first character of the returned string begins.
When position is 0 (zero), then it is treated as 1.
When position is positive, then the function counts from the beginning of string to find the first character.
<u>When position is negative</u>, then the function counts backward from the end of string.
substring_length
The length of the returned string. SUBSTR calculates lengths using characters as defined by the input character set. SUBSTRB uses bytes instead of characters. SUBSTRC uses Unicode complete characters. SUBSTR2 uses UCS2 code points. SUBSTR4 uses UCS4 code points.
When you do not specify a value for this argument, then the function

The **INSTR**(source string, search item, [start position],[nth occurrence of search item]) function returns a number that represents the position in the source string, beginning from the given start position, where the nth occurrence of the search item begins:
instr('http://www.domain.com','.',1,2) = 18

QUESTION 77
In the CUSTOMERS table, the CUST_CITY column contains the value 'Paris' for the CUST_FIRST_NAME 'ABIGAIL'.

Evaluate the following query:

```
SQL> SELECT INITCAP(cust_first_name || ' ' ||
UPPER(SUBSTR(cust_city,-LENGTH(cust_city),2)))
FROM customers
WHERE cust_first_name = 'ABIGAIL';
```

What would be the outcome?

A. Abigail PA
B. Abigail Pa
C. Abigail IS
D. an error message

Answer: B
Section: (none)

Explanation/Reference:

QUESTION 78
Evaluate the following query:

```
SQL> SELECT TRUNC(ROUND(156.00,-1),-1) FROM DUAL;
```

What would be the outcome?

A. 16
B. 100
C. 160
D. 200
E. 150

Answer: C
Section: (none)

Explanation/Reference:
Function Purpose
ROUND(**column|expression**, **n**) Rounds the column, expression, or value to **n** decimal places or, if **n** is omitted, no decimal places (If **n** is negative, numbers to the left of decimal point are rounded.)
TRUNC(**column|expression**, **n**) Truncates the column, expression, or value to **n** decimal places or, if **n** is omitted, **n** defaults to zero

QUESTION 79
View the Exhibit and examine the structure of the CUSTOMERS table.
In the CUSTOMERS table, the CUST_LAST_NAME column contains the values 'Anderson' and 'Ausson'.

You issue the following query:

```
SQL> SELECT LOWER(REPLACE(TRIM('son' FROM cust_last_name),'An','O'))
FROM CUSTOMERS
WHERE LOWER(cust_last_name) LIKE 'a%n';
```

What would be the outcome?

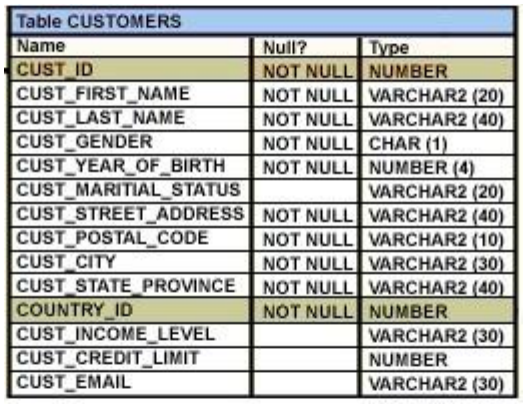


Table CUSTOMERS

Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

A. 'Oder' and 'Aus'
B. an error because the TRIM function specified is not valid
C. an error because the LOWER function specified is not valid
D. an error because the REPLACE function specified is not valid

Answer: B
Section: (none)

Explanation/Reference:
Function Purpose
ROUND(**column**|**expression**, **n**) Rounds the column, expression, or value to **n** decimal places or, if **n** is omitted, no decimal places (If **n** is negative, numbers to the left of decimal point are rounded.)
TRUNC(**column**|**expression**, **n**) Truncates the column, expression, or value to **n** decimal places or, if **n** is omitted, **n** defaults to zero

The TRIM Function
The TRIM function removes characters from the beginning or end of character literals, columns or expressions to yield one potentially shorter character item. Numeric and date literals are automatically cast as characters when they occur as parameters to the TRIM function. Numeric or date expressions are evaluated first before being converted to strings ready to be trimmed.
The TRIM function takes a parameter made up of an optional and a mandatory component. Its syntax is
TRIM([trailing|leading|both] trimstring from s).
The string to be trimmed (s) is mandatory. The following points list the rules governing the use
of this function:
■ TRIM(s) removes spaces from both sides of the input string.
■ TRIM(trailing trimstring from s) removes all occurrences of trimstring from the end of the string s if it is present.
■ TRIM(leading trimstring from s) removes all occurrences of trimstring from the beginning of the string s if it is present.
■ TRIM(both trimstring from s) removes all occurrences of trimstring from the beginning and end of the string s if it is present.
The following queries illustrate the usage of this function:
Query 1: select trim(trailing 'e' from 1+2.14||' is pie') from dual
Query 2: select trim(both '*' from '*******Hidden*******') from dual
Query 3: select trim(1 from sysdate) from dual

ORA-30001: trim set should have only one character
30001. 00000 - "trim set should have only one character"
***Cause: Trim set contains more or less than 1 character. This is not allowed in TRIM function.**

REPLACE(**text, search_string, replacement_string**)
Searches a text expression for a character string and, if found, replaces it with a specified replacement string

QUESTION 80
Which two statements are true regarding working with dates? (Choose two.)

A. The default internal storage of dates is in the numeric format.
B. The default internal storage of dates is in the character format.
C. The RR date format automatically calculates the century from the SYSDATE function and does not allow the user to enter the century.
D. The RR date format automatically calculates the century from the SYSDATE function but allows the user to enter the century if required.

Answer: AD
Section: (none)

Explanation/Reference:
Working with Dates

The Oracle Database stores dates in an internal numeric format, representing the century, year, month, day, hours, minutes, and seconds.
The default display and input format for any date is DD-MON-RR.

RR Date Format

The RR date format is similar to the YY element, but you can use it to specify different centuries. Use the RR date format element instead of YY so that the century of the return value varies according to the specified two-digit year and the last two digits of the current year. The table in the slide summarizes the behavior of the RR element.

Current Year	Given Date	Interpreted (RR)	Interpreted (YY)
1994	27-OCT-95	1995	1995
1994	27-OCT-17	2017	1917
2001	27-OCT-17	2017	2017
2048	27-OCT-52	1952	2052
2051	27-OCT-47	2147	2047

Note the values shown in the last two rows of the above table. As we approach the middle of the century, then the RR behavior is probably not what you want.
This data is stored internally as follows:
CENTURY YEAR MONTH DAY HOUR MINUTE SECOND 19 87 06 17 17 10 43

QUESTION 81
You are currently located in Singapore and have connected to a remote database in Chicago.

You issue the following command:

```
SQL> SELECT ROUND(SYSDATE-promo_begin_date,0)
FROM promotions
WHERE (SYSDATE-promo_begin_date)/365 > 2;
```

PROMOTIONS is the public synonym for the public database link for the PROMOTIONS table.

What is the outcome?

A. an error because the ROUND function specified is invalid
B. an error because the WHERE condition specified is invalid
C. number of days since the promo started based on the current Chicago date and time
D. number of days since the promo started based on the current Singapore date and time

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 82
Examine the data in the CUST_NAME column of the CUSTOMERS table.

```
CUST_NAME
------------------------
Renske Ladwig
Jason Mallin
Samuel McCain
Allan MCEwen
Irene Mikkilineni
Julia Nayer
```

You need to display customers' second names where the second name starts with "Mc" or "MC."

Which query gives the required output?

A. SELECT SUBSTR(cust_name, INSTR(cust_name,' ')+1)
   FROM customers
   WHERE INITCAP(SUBSTR(cust_name, INSTR(cust_name,' ')+1))='Mc';
B. SELECT SUBSTR(cust_name, INSTR(cust_name,' ')+1)
   FROM customers
   WHERE INITCAP(SUBSTR(cust_name, INSTR(cust_name,' ')+1)) LIKE 'Mc%';
C. SELECT SUBSTR(cust_name, INSTR(cust_name,' ')+1)
   FROM customers
   WHERE SUBSTR(cust_name, INSTR(cust_name,' ')+1) LIKE INITCAP('MC%');
D. SELECT SUBSTR(cust_name, INSTR(cust_name,' ')+1)
   FROM customers
   WHERE INITCAP(SUBSTR(cust_name, INSTR(cust_name,' ')+1)) = INITCAP('MC%');

Answer: B
Section: (none)

Explanation/Reference:

QUESTION 83
Examine the data in the CUST_NAME column of the CUSTOMERS table.

```
CUST_NAME
---------------------
Lex De Haan
Renske Ladwig
Jose Manuel Urman
Jason Mallin
```

You want to extract only those customer names that have three names and display the * symbol in place of the first name as follows:

```
CUST NAME
---------------------
*** De Haan
**** Manuel Urman
```

Which two queries give the required output? (Choose two.)

A. SELECT LPAD(SUBSTR(cust_name,INSTR(cust_name,' ')),LENGTH(cust_name),'*') "CUST NAME" FROM customers
   WHERE INSTR(cust_name, ' ',1,2)<>0;
B. SELECT LPAD(SUBSTR(cust_name,INSTR(cust_name,' ')),LENGTH(cust_name),'*') "CUST NAME" FROM customers
   WHERE INSTR(cust_name, ' ',-1,2)<>0;
C. SELECT LPAD(SUBSTR(cust_name,INSTR(cust_name,' ')),LENGTH(cust_name)- INSTR(cust_name,''),'*') "CUST NAME"
   FROM customers
   WHERE INSTR(cust_name, ' ',-1,-2)<>0;
D. SELECT LPAD(SUBSTR(cust_name,INSTR(cust_name,' ')),LENGTH(cust_name)- INSTR(cust_name,' '),'*') "CUST NAME"
   FROM customers
   WHERE INSTR(cust_name, ' ',1,2)<>0 ;

Answer: AB
Section: (none)

Explanation/Reference:

QUESTION 84
View the Exhibit and examine the structure of the EMPLOYEES table.

Examine the data in the ENAME and HIREDATE columns of the EMPLOYEES table:

```
ENAME    HIREDATE
---------------------------------
SMITH 17-DEC-80
ALLEN 20-FEB-81
WARD 22-FEB-81
```

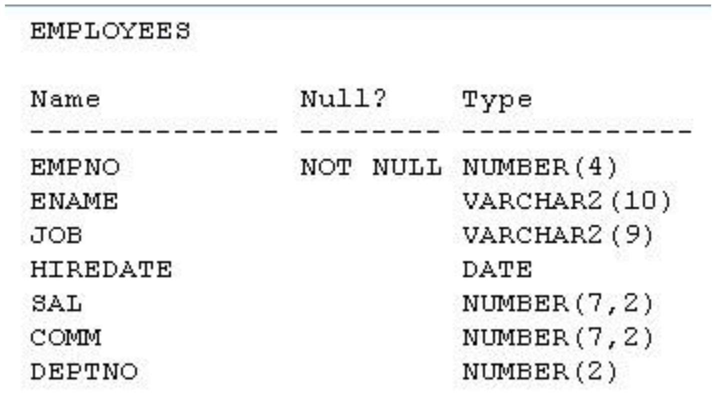
You want to generate a list of user IDs as follows:

```
USERID
-------------
Smi17DEC80
All20FEB81
War22FEB81
```

You issue the following query:

```
SQL>SELECT CONCAT(SUBSTR(INITCAP(ename),1,3), REPLACE(hiredate,'-')) "USERID"
FROM employees;
```

What is the outcome?



```
EMPLOYEES

Name            Null?     Type
--------------- --------- -------------
EMPNO           NOT NULL  NUMBER(4)
ENAME                     VARCHAR2(10)
JOB                       VARCHAR2(9)
HIREDATE                  DATE
SAL                       NUMBER(7,2)
COMM                      NUMBER(7,2)
DEPTNO                    NUMBER(2)
```

A. It executes successfully and gives the correct output.
B. It executes successfully but does not give the correct output.
C. It generates an error because the REPLACE function is not valid.
D. It generates an error because the SUBSTR function cannot be nested in the CONCAT function.

Answer: A
Section: (none)

Explanation/Reference:
REPLACE(**text, search_string,replacement_string**)
Searches a text expression for a character string and, if found, replaces it with a specified replacement string

The REPLACE Function

The REPLACE function replaces all occurrences of a search item in a source string with a replacement term and returns the modified source string. If the length of the replacement term is different from that of the search item, then the lengths of the returned and source strings will be different. If the search string is not found, the source string is returned unchanged. Numeric and date literals and expressions are evaluated before being implicitly cast as characters when they occur as parameters to the REPLACE function.

The REPLACE function takes three parameters, with the first two being mandatory. Its syntax is REPLACE(source string, search item, [replacement term]).
I**f the replacement term parameter is omitted, each occurrence of the search item is removed from the source string.** In other words, the search item is replaced by an empty string. .
The following queries illustrate the REPLACE function with numeric
and date expressions:
Query 1: select replace(10000-3,'9','85') from dual
Query 2: select replace(sysdate, 'DEC','NOV') from dual

QUESTION 85
View the Exhibit and examine the structure and data in the INVOICE table.

```
name Null Type
---------------------------
INV_NO NOT NULL NUMBER(3)
INV_DATE DATE
INV_AMT NUMBER(10,2)
```

Which statements are true regarding data type conversion in expressions used in queries? (Choose all that apply.)

A. inv_amt ='0255982' : requires explicit conversion
B. inv_date > '01-02-2008' : uses implicit conversion
C. CONCAT(inv_amt,inv_date) : requires explicit conversion
D. inv_date = '15-february-2008' : uses implicit conversion
E. inv_no BETWEEN '101' AND '110' : uses implicit conversion

Answer: DE
Section: (none)

Explanation/Reference:
In some cases, the Oracle server receives data of one data type where it expects data of a different data type. When this happens, the Oracle server can automatically convert the data to the expected data type. This data type conversion can be done **implicitly** by the Oracle server or **explicitly** by the user.
Explicit data type conversions are performed by using the conversion functions. Conversion functions convert a value from one data type to another. Generally, the form of the function names follows the convention **data type** TO **data type**. The first data type is the input data type and the second data type is the output.
Note: Although implicit data type conversion is available, it is recommended that you do the explicit data type conversion to ensure the reliability of your SQL statements.

QUESTION 86
Examine the structure and data of the CUST_TRANS table:

```
CUST_TRANS
-------------------------
Name Null Type
CUSTNO NOT NULL CHAR(2)
TRANSDATE DATE
TRANSAMT NUMBER(6,2)
```

```
CUSTNO TRANSDATE TRANSAMT
---------------------------------------
11 01-JAN-07 1000
22 01-FEB-07 2000
33 01-MAR-07 3000
```

Dates are stored in the default date format dd-mon-rr in the CUST_TRANS table.

Which SQL statements would execute successfully? (Choose three .)

A. SELECT transdate + '10' FROM cust_trans;
B. SELECT * FROM cust_trans WHERE transdate = '01-01-07';
C. SELECT transamt FROM cust_trans WHERE custno > '11';
D. SELECT * FROM cust_trans WHERE transdate='01-JANUARY-07';
E. SELECT custno + 'A' FROM cust_trans WHERE transamt > 2000;

Answer: ACD
Section: (none)

Explanation/Reference:

QUESTION 87
You want to display the date for the first Monday of the next month and issue the following command:

```
SQL>SELECT TO_CHAR(NEXT_DAY(LAST_DAY(SYSDATE),'MON'), 'dd "is the first Monday for" fmmonth rrrr')
FROM DUAL;
```

What is the outcome?

A. It executes successfully and returns the correct result.
B. It executes successfully but does not return the correct result.
C. It generates an error because TO_CHAR should be replaced with TO_DATE.
D. It generates an error because rrrr should be replaced by rr in the format string.
E. It generates an error because fm and double quotation marks should not be used in the format string.

Answer: A
Section: (none)

Explanation/Reference:
• NEXT_DAY(date, 'char'): Finds the date of the next specified day of the week ('char') following date. The value of char may be a number representing a day or a character string.
• LAST_DAY(date): Finds the date of the last day of the month that contains date
The second innermost function is evaluated next. TO_CHAR('28-OCT-2009', 'fmMonth') converts the given date based on the Month format mask and returns the character string October. The fm modifier trims trailing blank spaces from the name of the month.

QUESTION 88
You need to calculate the number of days from 1st January 2007 till date.

Dates are stored in the default format of **dd-mon-rr**.

Which SQL statements would give the required output? (Choose two .)

A. SELECT SYSDATE - '01-JAN-2007' FROM DUAL;
B. SELECT SYSDATE - TO_DATE('01/JANUARY/2007') FROM DUAL;
C. SELECT SYSDATE - TO_DATE('01-JANUARY-2007') FROM DUAL;
D. SELECT TO_CHAR(SYSDATE, 'DD-MON-YYYY') - '01-JAN-2007' FROM DUAL;
E. SELECT TO_DATE(SYSDATE, 'DD/MONTH/YYYY') - '01/JANUARY/2007' FROM DUAL;

Answer: BC
Section: (none)

Explanation/Reference:

QUESTION 89

You need to display the date 11-oct-2007 in words as 'Eleventh of October, Two Thousand Seven'.

Which SQL statement would give the required result?

A. SELECT TO_CHAR('11-oct-2007', 'fmDdspth "of" Month, Year') FROM DUAL;
B. SELECT TO_CHAR(TO_DATE('11-oct-2007'), 'fmDdspth of month, year') FROM DUAL;
C. SELECT TO_CHAR(TO_DATE('11-oct-2007'), 'fmDdthsp "of" Month, Year') FROM DUAL;
D. SELECT TO_DATE(TO_CHAR('11-oct-2007','fmDdspth "of" Month, Year')) FROM DUAL;

Answer: C
Section: (none)

Explanation/Reference:
Using the **TO_CHAR** Function with Dates
TO_CHAR converts a datetime data type to a value of VARCHAR2 data type in the format specified by the format_model. A format model is a character literal that describes the format of datetime stored in a character string. For example, the datetime format model for the string '11-Nov-1999' is 'DD-Mon-YYYY'. You can use the TO_CHAR function to convert a date from its default format to the one that you specify.
Guidelines
• The format model must be enclosed with single quotation marks and is case-sensitive.
• The format model can include any valid date format element. But be sure to separate the
date value from the format model with a comma.
• The names of days and months in the output are automatically padded with blanks.
• To remove padded blanks or to suppress leading zeros, use the fill mode **fm** element.

Elements of the Date Format Model

--
DY Three-letter abbreviation of the day of the week
DAY Full name of the day of the week
DD Numeric day of the month

MM Two-digit value for the month
MON Three-letter abbreviation of the month
MONTH Full name of the month

YYYY Full year in numbers
YEAR Year spelled out (in English)

QUESTION 90
Examine the structure and data in the PRICE_LIST table:

name Null Type

PROD_ID NOT NULL NUMBER(3)
PROD_PRICE VARCHAR2(10)

PROD_ID PROD_PRICE
----
100 $234.55
101 $6,509.75
102 $1,234

You plan to give a discount of 25% on the product price and need to display the discount amount in the same format as the PROD_PRICE.

Which SQL statement would give the required result?

A. SELECT TO_CHAR(prod_price* .25,'$99,999.99')
   FROM PRICE_LIST;
B. SELECT TO_CHAR(TO_NUMBER(prod_price)* .25,'$99,999.00')

FROM PRICE_LIST;

C. SELECT TO_CHAR(TO_NUMBER(prod_price,'$99,999.99')* .25,'$99,999.00') FROM PRICE_LIST;
D. SELECT TO_NUMBER(TO_NUMBER(prod_price,'$99,999.99')* .25,'$99,999.00') FROM PRICE_LIST;

Answer: C
Section: (none)

Explanation/Reference:
Using the TO_CHAR Function
The TO_CHAR function returns an item of data type VARCHAR2. When applied to items of type NUMBER, several formatting options are available. The syntax is as follows:
TO_CHAR(number1, [format], [nls_parameter]),
The number1 parameter is mandatory and must be a value that either is or can be implicitly converted into a number. The optional format parameter may be used to specify numeric formatting information like width, currency symbol, the position of a decimal point, and group (or thousands) separators and must be enclosed in single

Syntax of Explicit Data Type Conversion
Functions
TO_NUMBER(char1, [format mask], [nls_parameters]) = num1
TO_CHAR(num1, [format mask], [nls_parameters]) = char1
TO_DATE(char1, [format mask], [nls_parameters]) = date1
TO_CHAR(date1, [format mask], [nls_parameters]) = char1

QUESTION 91
View the Exhibit and examine the structure of the PROMOTIONS table.

Which two SQL statements would execute successfully? (Choose two.)

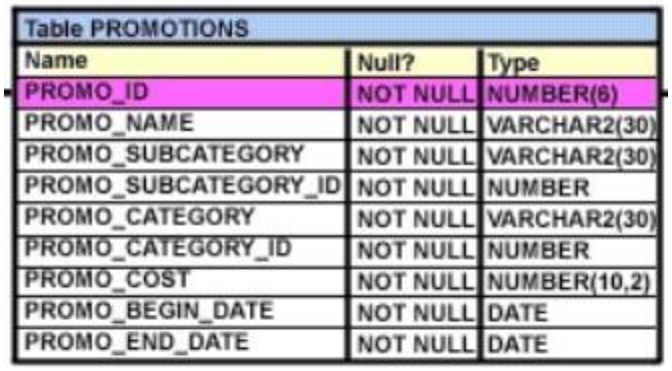


Table PROMOTIONS

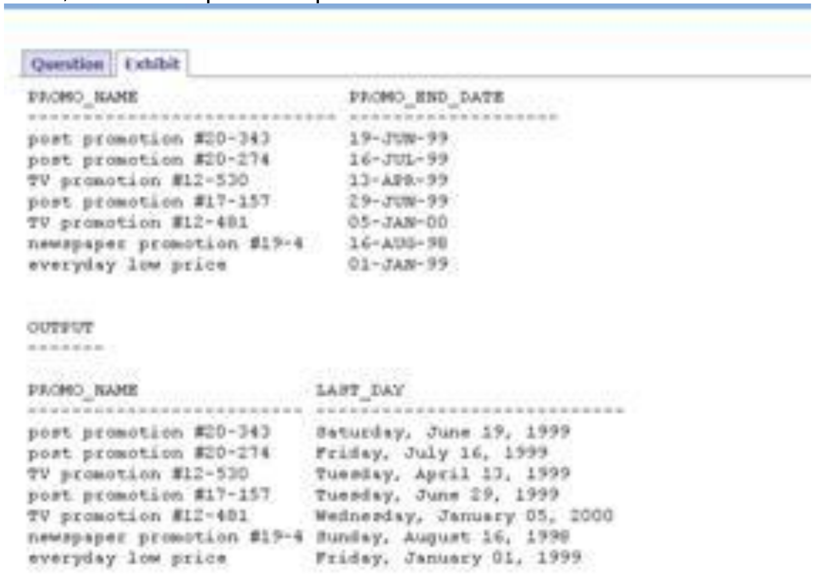
Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

A. UPDATE promotions
SET promo_cost = promo_cost+ 100
WHERE TO_CHAR(promo_end_date, 'yyyy') > '2000';
B. SELECT promo_begin_date
FROM promotions
WHERE TO_CHAR(promo_begin_date,'mon dd yy')='jul 01 98';
C. UPDATE promotions
SET promo_cost = promo_cost+ 100
WHERE promo_end_date > TO_DATE(SUBSTR('01-JAN-2000',8));
D. SELECT TO_CHAR(promo_begin_date,'dd/month')
FROM promotions
WHERE promo_begin_date IN (TO_DATE('JUN 01 98'), TO_DATE('JUL 01 98'));

Answer: AB
Section: (none)

Explanation/Reference:

QUESTION 92
View the Exhibit and examine the data in the PROMO_NAME and PROMO_END_DATE columns of the PROMOTIONS table, and the required output format.



```
PROMO_NAME                      PROMO_END_DATE
------------------------------- ------------------
post promotion #20-343          19-JUN-99
post promotion #20-274          16-JUL-99
TV promotion #12-530            13-APR-99
post promotion #17-157          29-JUN-99
TV promotion #12-481            05-JAN-00
newspaper promotion #19-4       16-AUG-98
everyday low price              01-JAN-99


OUTPUT
-------

PROMO_NAME                  LAST_DAY
--------------------------- ----------------------------
post promotion #20-343      Saturday, June 19, 1999
post promotion #20-274      Friday, July 16, 1999
TV promotion #12-530        Tuesday, April 13, 1999
post promotion #17-157      Tuesday, June 29, 1999
TV promotion #12-481        Wednesday, January 05, 2000
newspaper promotion #19-4 Sunday, August 16, 1998
everyday low price          Friday, January 01, 1999
```

Which two queries give the correct result? (Choose two.)

A. SELECT promo_name, TO_CHAR(promo_end_date,'Day') ', '
   TO_CHAR(promo_end_date,'Month') ' '
   TO_CHAR(promo_end_date,'DD, YYYY') AS last_day
   FROM promotions;
B. SELECT promo_name,TO_CHAR (promo_end_date,'fxDay') ', '
   TO_CHAR(promo_end_date,'fxMonth') ' '
   TO_CHAR(promo_end_date,'fxDD, YYYY') AS last_day
   FROM promotions;
C. SELECT promo_name, TRIM(TO_CHAR(promo_end_date,'Day')) ', ' TRIM(TO_CHAR(promo_end_date,'Month')) ' '
   TRIM(TO_CHAR(promo_end_date,'DD, YYYY')) AS last_day
   FROM promotions;
D. SELECTpromo_name,TO_CHAR(promo_end_date,'fmDay')','
   TO_CHAR(promo_end_date,'fmMonth') ' '
   TO_CHAR(promo_end_date,'fmDD, YYYY') AS last_day
   FROM promotions;

Answer: CD
Section: (none)

Explanation/Reference:

QUESTION 93
View the Exhibit and examine the structure of the CUSTOMERS table.

Using the CUSTOMERS table, y ou need to generate a report that shows an increase in the credit limit by 15% for all customers.
Customers whose credit limit has not been entered should have the message " Not Available" displayed.

Which SQL statement would produce the required result?

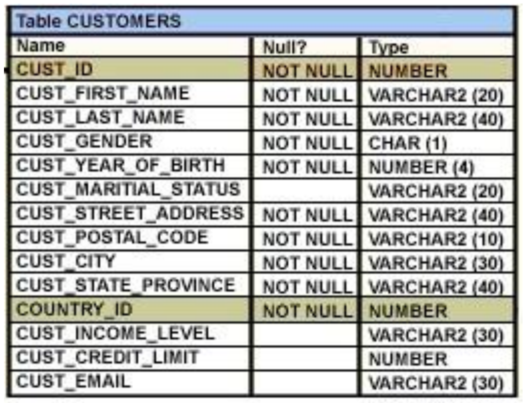


Table CUSTOMERS		
Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

A. SELECT NVL(cust_credit_limit,'Not Available')*.15 "NEW CREDIT" FROM customers;
B. SELECT NVL(cust_credit_limit*.15,'Not Available') "NEW CREDIT" FROM customers;
C. SELECT TO_CHAR(NVL(cust_credit_limit*.15,'Not Available')) "NEW CREDIT" FROM customers;
D. SELECT NVL(TO_CHAR(cust_credit_limit*.15),'Not Available') "NEW CREDIT" FROM customers;

Answer: D
Section: (none)

Explanation/Reference:
NVL Function
Converts a null value to an actual value:
 Data types that can be used are date, character, and number.
 Data types must match:
– NVL(commission_pct,0)
– NVL(hire_date,'01-JAN-97')
– NVL(job_id,'No Job Yet')

QUESTION 94
Examine the structure of the PROGRAMS table:

name Null Type
PROG_ID NOT NULL NUMBER(3)
PROG_COST NUMBER(8,2)
START_DATE NOT NULL DATE
END_DATE DATE

Which two SQL statements would execute successfully? (Choose two.)

A. SELECT NVL(ADD_MONTHS(END_DATE,1),SYSDATE)
FROM programs;
B. SELECT TO_DATE(NVL(SYSDATE-END_DATE,SYSDATE))
FROM programs;
C. SELECT NVL(MONTHS_BETWEEN(start_date,end_date),'Ongoing')
FROM programs;
D. SELECT NVL(TO_CHAR(MONTHS_BETWEEN(start_date,end_date)),'Ongoing') FROM programs;

Answer: AD
Section: (none)

Explanation/Reference:
NVL Function

Converts a null value to an actual value:
 Data types that can be used are date, character, and number.
 Data types <u>must match:</u>
– NVL(commission_pct,0)
– NVL(hire_date,'01-JAN-97')
– NVL(job_id,'No Job Yet')

MONTHS_BETWEEN(**date1, date2**): Finds the number of months between **date1** and **date2**. The result can be positive or negative. If **date1** is later than **date2**, the
result is positive; if **date1** is earlier than **date2**, the result is negative. The noninteger part of the result represents a portion of the month.
MONTHS_BETWEEN returns a **numeric value**. - answer C NVL has different datatypes - numeric and strings, which is not possible!

The data types of the original and ifnull parameters must always be compatible. They must either be of the same type, or it must be possible to implicitly convert ifnull to the type of the original parameter. The NVL function returns a value with the same data type as the original parameter.

QUESTION 95
The PRODUCTS table has the following structure:

```
name Null Type
PROD_ID NOT NULL NUMBER(4)
PROD_NAME VARCHAR2(25)
PROD_EXPIRY_DATE DATE
```

Evaluate the following two SQL statements:

```
SQL>SELECT prod_id, NVL2(prod_expiry_date, prod_expiry_date + 15,'')
FROM products;

SQL>SELECT prod_id, NVL(prod_expiry_date, prod_expiry_date + 15)
FROM products;
```

Which statement is true regarding the outcome?

A. Both the statements execute and give different results.
B. Both the statements execute and give the same result.
C. Only the first SQL statement executes successfully.
D. Only the second SQL statement executes successfully.

Answer: A
Section: (none)

Explanation/Reference:
Using the NVL2 Function
The NVL2 function examines the first expression. If the first expression is not null, the NVL2 function returns the second expression. If the first expression is null, the third expression is returned.
Syntax
NVL2(**expr1**, **expr2, expr3**)
In the syntax:
 expr1 is the source value or expression that may contain a null
 expr2 is the value that is returned if **expr1** is not null
 expr3 is the value that is returned if **expr1** is null

QUESTION 96
Examine the structure of the INVOICE table.

name Null Type
INV_NO NOT NULL NUMBER(3)
INV_DATE DATE
INV_AMT NUMBER(10,2)

Which two SQL statements would execute successfully? (Choose two.)

A. SELECT inv_no,NVL2(inv_date,'Pending','Incomplete')
   FROM invoice;
B. SELECT inv_no,NVL2(inv_amt,inv_date,'Not Available')
   FROM invoice;
C. SELECT inv_no,NVL2(inv_date,sysdate-inv_date,sysdate)
   FROM invoice;
D. SELECT inv_no,NVL2(inv_amt,inv_amt*.25,'Not Available')
   FROM invoice;

Answer: AC
Section: (none)

Explanation/Reference:
The NVL2 Function
The NVL2 function provides an enhancement to NVL but serves a very similar purpose. It evaluates whether a column or expression of any data type is null or not.
5-6 The NVL function
If the first term is not null, the second parameter is returned, else the third parameter is returned. Recall that the NVL function is different since it returns the original term if it is not null. The NVL2 function takes three mandatory parameters. Its syntax is NVL2(original, ifnotnull, ifnull), where original represents the term being tested. Ifnotnull is returned if original is not null, and ifnull is returned if original is null. The data types of the ifnotnull and ifnull parameters must be compatible, and they cannot be of type LONG.
They must either be of the same type, or it must be possible to convert ifnull to the type of the ifnotnull parameter. The data type returned by the NVL2 function is the same as that of the **ifnotnull** parameter.

QUESTION 97
View the Exhibit and evaluate the structure and data in the CUST_STATUS table.

You issue the following SQL statement:

```
SQL> SELECT custno, NVL2(NULLIF(amt_spent, credit_limit), 0, 1000)"BONUS"
FROM cust_status;
```

Which statement is true regarding the execution of the above query?

A. It produces an error because the AMT_SPENT column contains a null value.
B. It displays a bonus of 1000 for all customers whose AMT_SPENT is less than CREDIT_LIMIT.
C. It displays a bonus of 1000 for all customers whose AMT_SPENT equals CREDIT_LIMIT, or AMT_SPENT is null .
D. It produces an error because the TO_NUMBER function must be used to convert the result of the NULLIF function before it can be used by the NVL2 function.

Answer: C
Section: (none)

Explanation/Reference:
The NULLIF Function

The NULLIF function tests two terms for equality. If they are equal the function returns a null, else it returns the first of the two terms tested.
The NULLIF function takes two mandatory parameters of any data type. The syntax is NULLIF(ifunequal, comparison_term), where the parameters ifunequal and
comparison_term are compared. If they are identical, then NULL is returned. If they
differ, the ifunequal parameter is returned.

QUESTION 98
Which statement is true regarding the COALESCE function?

A. It can have a maximum of five expressions in a list.
B. It returns the highest NOT NULL value in the list for all rows.
C. It requires that all expressions in the list must be of the same data type.
D. It requires that at least one of the expressions in the list must have a NOT NULL value.

Answer: C
Section: (none)

Explanation/Reference:
The COALESCE Function
The COALESCE function returns the first nonnull value from its parameter list. If all its parameters are null, then null is returned.
The COALESCE function takes two mandatory parameters and any number of optional parameters. The syntax is COALESCE(expr1, expr2,…,exprn), where expr1 is returned if it is not null, else expr2 if it is not null, and so on.
COALESCE is a general form of the NVL function, as the following two equations illustrate:
COALESCE(expr1,expr2) = NVL(expr1,expr2)
COALESCE(expr1,expr2,expr3) = NVL(expr1,NVL(expr2,expr3))
The data type COALESCE returns if a not null value is found is the same as that of the first not null parameter. To avoid an "ORA-00932: inconsistent data types" error, all not null parameters must have data types compatible with the first not null parameter.

QUESTION 99
View the Exhibit and examine the structure of the PROMOTIONS table.

Using the PROMOTIONS table, you need to find out the average cost for all promos in the ranges $0-2000 and $2000-5000 in category A

You issue the following SQL statement:

```
SQL>SELECT AVG(CASE
WHEN promo_cost BETWEEN 0 AND 2000 AND promo_category='A'
then promo_cost
ELSE null END) "CAT_2000A",
AVG(CASE
WHEN promo_cost BETWEEN 2001 AND 5000 AND promo_category='A'
THEN promo_cost
ELSE null END) "CAT_5000A"
FROM promotions;
```

What would be the outcome?

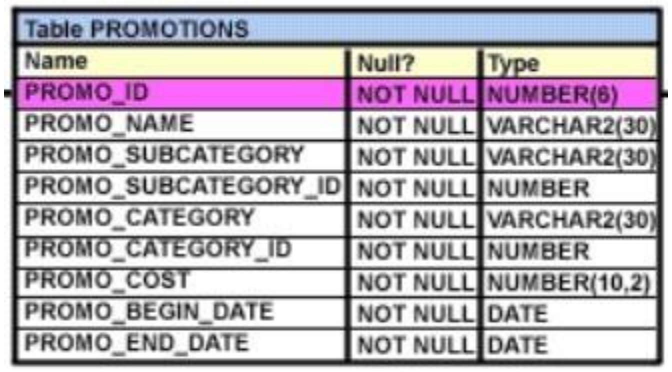


Table PROMOTIONS		
Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

A. It executes successfully and gives the required result.
B. It generates an error because NULL cannot be specified as a return value.
C. It generates an error because CASE cannot be used with group functions.
D. It generates an error because multiple conditions cannot be specified for the WHEN clause.

Answer: A
Section: (none)

Explanation/Reference:
CASE Expression
Facilitates conditional inquiries by doing the work of an
IF-THEN-ELSE statement:
CASE expr WHEN comparison_expr1 THEN return_expr1
[WHEN comparison_expr2 THEN return_expr2
WHEN comparison_exprn THEN return_exprn
ELSE else_expr]
END

QUESTION 100
View the Exhibit and examine the structure of the PROMOTIONS table.

Which SQL statements are valid? (Choose all that apply.)

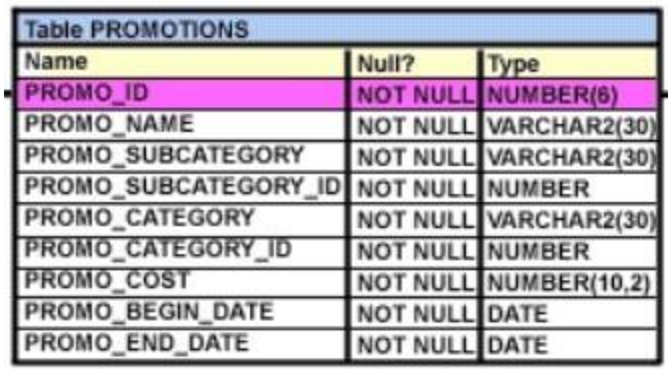


Table PROMOTIONS		
Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

A. SELECT promo_id, DECODE(NVL(promo_cost,0), promo_cost,
promo_cost * 0.25, 100) "Discount"
FROM promotions;
B. SELECT promo_id, DECODE(promo_cost, 10000,
DECODE(promo_category, 'G1', promo_cost *.25, NULL),
NULL) "Catcost"
FROM promotions;
C. SELECT promo_id, DECODE(NULLIF(promo_cost, 10000),

NULL, promo_cost*.25, 'N/A') "Catcost"
FROM promotions;

D. SELECT promo_id, DECODE(promo_cost, >10000, 'High',
<10000, 'Low') "Range"
FROM promotions;

Answer: AB
Section: (none)

Explanation/Reference:
The DECODE Function
Although its name sounds mysterious, this function is straightforward. The DECODE function implements if-then-else conditional logic by testing its first two terms for **equality** and returns the third if they are equal and optionally returns another term if they are not.
The DECODE function takes at least three mandatory parameters, but can
take many more. The syntax of the function is DECODE(expr1,comp1, iftrue1,
[comp2,iftrue2...[compN,iftrueN]], [iffalse]).

QUESTION 101
Examine the data in the PROMO_BEGIN_DATE column of the PROMOTIONS table:

```
PROMO_BEGIN _DATE
-----------------------------------
04-jan-00
10-jan-00
15-dec-99
18-oct-98
22-aug-99
```

You want to display the number of promotions started in 1999 and 2000.

Which query gives the correct output?

A. SELECT SUM(DECODE(SUBSTR(promo_begin_date,8),'00',1,0)) "2000",
SUM(DECODE(SUBSTR(promo_begin_date,8),'99',1,0)) "1999"
FROM promotions;

B. SELECT SUM(CASE TO_CHAR(promo_begin_date,'yyyy') WHEN '99' THEN 1 ELSE 0 END) "1999",SUM(CASE TO_CHAR(promo_begin_date,'yyyy') WHEN '00' THEN 1 ELSE 0 END) "2000"
FROM promotions;

C. SELECT COUNT(CASE TO_CHAR(promo_begin_date,'yyyy') WHEN '99' THEN 1 ELSE 0 END) "1999",COUNT(CASE TO_CHAR(promo_begin_date,'yyyy') WHEN '00' THEN 1 ELSE 0 END) "2000"
FROM promotions;

D. SELECT COUNT(DECODE(SUBSTR(TO_CHAR(promo_begin_date,'yyyy'), 8), '1999', 1, 0)) "1999",
COUNT(DECODE(SUBSTR(TO_CHAR(promo_begin_date,'yyyy'), 8),'2000', 1,
0)) "2000"
FROM promotions;

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 102
Examine the structure of the TRANSACTIONS table:

```
name              Null              Type
------------------------------------------------------
TRANS_ID       NOT NULL    NUMBER(3)
```

```
CUST_NAME                    VARCHAR2(30)
TRANS_DATE                   TIMESTAMP
TRANS_AMT                    NUMBER(10,2)
```

You want to display the date, time, and transaction amount of transactions that where done before 12 noon. The value zero should be displayed for transactions where the transaction amount has not been entered.

Which query gives the required result?

A. SELECT TO_CHAR(trans_date,'dd-mon-yyyy hh24:mi:ss'),
TO_CHAR(trans_amt,'$99999999D99')
FROM transactions
WHERE TO_NUMBER(TO_DATE(trans_date,'hh24')) < 12 AND COALESCE(trans_amt,NULL)<>NULL;
B. SELECT TO_CHAR(trans_date,'dd-mon-yyyy hh24:mi:ss'),
NVL(TO_CHAR(trans_amt,'$99999999D99'),0)
FROM transactions
WHERE TO_CHAR(trans_date,'hh24') < 12;
C. SELECT TO_CHAR(trans_date,'dd-mon-yyyy hh24:mi:ss'),
COALESCE(TO_NUMBER(trans_amt,'$99999999.99'),0)
FROM transactions
WHERE TO_DATE(trans_date,'hh24') < 12;
D. SELECT TO_DATE (trans_date,'dd-mon-yyyy hh24:mi:ss'),
NVL2(trans_amt,TO_NUMBER(trans_amt,'$99999999.99'), 0)
FROM transactions
WHERE TO_DATE(trans_date,'hh24') < 12;

Answer: B
Section: (none)

Explanation/Reference:

QUESTION 103
Examine the structure of the TRANSACTIONS table:

```
name Null Type
TRANS_ID NOT NULL NUMBER(3)
CUST_NAME VARCHAR2(30)
TRANS_DATE DATE
TRANS_AMT NUMBER(10,2)
```

You want to display the transaction date and specify whether it is a weekday or weekend.

Evaluate the following two queries:

```
SQL>SELECT TRANS_DATE,CASE
WHEN TRIM(TO_CHAR(trans_date,'DAY')) IN ('SATURDAY','SUNDAY') THEN 'weekend'
ELSE 'weekday'
END "Day Type"
FROM transactions;

SQL>SELECT TRANS_DATE, CASE
WHEN TO_CHAR(trans_date,'DAY') BETWEEN 'MONDAY' AND 'FRIDAY' THEN 'weekday'
ELSE 'weekend'
END "Day Type"FROM transactions;
```

Which statement is true regarding the above queries?

A. Both give wrong results.
B. Both give the correct result.

C. Only the first query gives the correct result.
D. Only the second query gives the correct result.

Answer: C
Section: (none)

Explanation/Reference:
Range Conditions Using the BETWEEN Operator
Use the BETWEEN operator to display rows based on a range of values:

```
SELECT last_name, salary
FROM employees
WHERE salary BETWEEN 2500 AND 3500 ;
```

Range Conditions Using the BETWEEN Operator
You can display rows based on a range of values using the BETWEEN operator. The range that you specify contains a lower limit and an upper limit.
The SELECT statement in the slide returns rows from the EMPLOYEES table for any employee whose salary is between $2,500 and $3,500.
Values that are specified with the BETWEEN operator are inclusive. However, you must specify the lower limit first.
You can also use the BETWEEN operator on character values:

```
SELECT last_name
FROM employees
WHERE last_name BETWEEN 'King' AND 'Smith';
```

QUESTION 104
Examine the structure of the PROMOS table:

```
name Null Type
PROMO_ID NOT NULL NUMBER(3)
PROMO_NAME VARCHAR2(30)
PROMO_START_DATE NOT NULL DATE
PROMO_END_DATE DATE
```

You want to generate a report showing promo names and their duration (number of days).
If the PROMO_END_DATE has not been entered, the message 'ONGOING' should be displayed.

Which queries give the correct output? (Choose all that apply.)

A. SELECT promo_name, TO_CHAR(NVL(promo_end_date -promo_start_date,'ONGOING')) FROM promos;
B. SELECT promo_name,COALESCE(TO_CHAR(promo_end_date - promo_start_date),'ONGOING') FROM promos;
C. SELECT promo_name, NVL(TO_CHAR(promo_end_date -promo_start_date),'ONGOING') FROM promos;
D. SELECT promo_name, DECODE(promo_end_date
-promo_start_date,NULL,'ONGOING',promo_end_date - promo_start_date) FROM promos;
E. SELECT promo_name, decode(coalesce(promo_end_date,promo_start_date),null,'ONGOING', promo_end_date - promo_start_date)
FROM promos;

Answer: BCD
Section: (none)

Explanation/Reference:

QUESTION 105
Examine the structure of the PROMOS table:

```
name                                    Null            Type
-----------------------------------------------------------------------------
PROMO_ID                            NOT NULL    NUMBER(3)
PROMO_NAME                                          VARCHAR2(30)
```

PROMO_START_DATE NOT NULL DATE
PROMO_END_DATE NOT NULL DATE

You want to display the list of promo names with the message 'Same Day' for promos that started and ended on the same day.

Which query gives the correct output?

A. SELECT promo_name, NVL(NULLIF(promo_start_date, promo_end_date), 'Same Day') FROM promos;
B. SELECT promo_name, NVL(TRUNC(promo_end_date - promo_start_date), 'Same Day') FROM promos;
C. SELECT promo_name, NVL2(TO_CHAR(TRUNC(promo_end_date-promo_start_date)), NULL,'Same Day')
FROM promos;
D. SELECT promo_name, DECODE((NULLIF(promo_start_date, promo_end_date)), NULL,'Same day') FROM promos;

Answer: D
Section: (none)

Explanation/Reference:

The NULLIF Function

The NULLIF function tests two terms for equality. If they are equal the function returns a null, else it returns the first of the two terms tested.
The NULLIF function takes two mandatory parameters of any data type. The syntax is NULLIF(ifunequal, comparison_term), where the parameters ifunequal and comparison_term are compared. If they are identical, then NULL is returned. If they differ, the ifunequal parameter is returned

ANSWER A - date and String incompatibl;a datatypes for NVL function

The Date TRUNC Function

The date TRUNC function performs a truncation operation on a date value based on a specified date precision format.
The date TRUNC function takes one mandatory and one optional parameter.
Its syntax is TRUNC(source date, [date precision format]). The source date parameter represents any value that can be implicitly converted into a date item. The date precision format parameter specifies the degree of truncation and is optional. If it is absent, the default degree of truncation is day. This means that any time component

QUESTION 106
Examine the data in the LIST_PRICE and MIN_PRICE columns of the PRODUCTS table:

```
LIST_PRICE MIN_PRICE
-------------------------------------
10000 8000
20000
30000 30000
```

Which two expressions give the same output? (Choose two.)

A. NVL(NULLIF(list_price, min_price), 0)
B. NVL(COALESCE(list_price, min_price), 0)
C. NVL2(COALESCE(list_price, min_price), min_price, 0)
D. COALESCE(NVL2(list_price, list_price, min_price), 0)

Answer: BD
Section: (none)

Explanation/Reference:

Using the COALESCE Function

• The advantage of the COALESCE function over the NVL function is that the COALESCE function can take multiple alternate values.
• If the first expression is not null, the COALESCE function returns that expression; otherwise, it does a COALESCE of the remaining expressions.

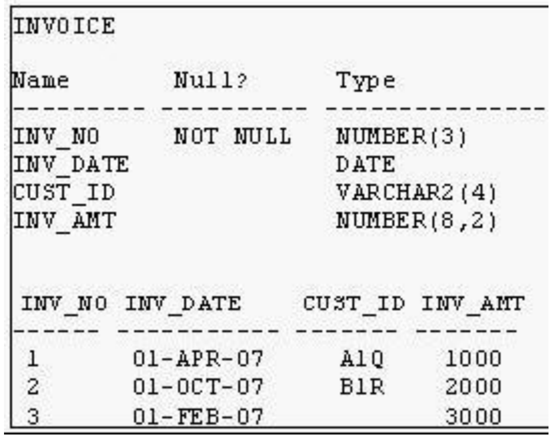
Using the COALESCE Function
The COALESCE function returns the first non-null expression in the list.
Syntax

COALESCE (**expr1**, **expr2, ... exprn**)

In the syntax:
- **expr1** returns this expression if it is not null
- **expr2** returns this expression if the first expression is null and this expression is not null
- **exprn** returns this expression if the preceding expressions are null

Note that all expressions must be of the same data type.

QUESTION 107
View the Exhibit and examine the structure and data in the INVOICE table.



```
INVOICE

Name      Null?      Type
--------- ---------- ---------------
INV_NO    NOT NULL   NUMBER(3)
INV_DATE             DATE
CUST_ID              VARCHAR2(4)
INV_AMT              NUMBER(8,2)


 INV_NO INV_DATE    CUST_ID INV_AMT
------ ----------- ------- -------
 1      01-APR-07    A1Q     1000
 2      01-OCT-07    B1R     2000
 3      01-FEB-07            3000
```

Which two SQL statements would execute successfully? (Choose two.)

A. SELECT AVG(inv_date)
   FROM invoice;
B. SELECT MAX(inv_date),MIN(cust_id)
   FROM invoice;
C. SELECT MAX(AVG(SYSDATE - inv_date))
   FROM invoice;
D. SELECT AVG(inv_date - SYSDATE), AVG(inv_amt)
   FROM invoice;

Answer: BD
Section: (none)

Explanation/Reference:
Using the AVG and SUM Functions
You can use the AVG, SUM, MIN, and MAX functions against the columns that can store numeric data.
The example in the slide displays the average, highest, lowest, and sum of monthly salaries for all sales representatives
Using the MIN and MAX Functions
You can use the MAX and MIN functions for numeric, character, and date data types. The example in the slide displays the most junior and most senior employees.

QUESTION 108
Which two statements are true regarding the COUNT function? (Choose two.)

A. The COUNT function can be used only for CHAR, VARCHAR2, and NUMBER data types.
B. COUNT(*) returns the number of rows including duplicate rows and rows containing NULL value in any of the columns.
C. COUNT(cust_id) returns the number of rows including rows with duplicate customer IDs and NULL value in the CUST_ID column.
D. COUNT(DISTINCT inv_amt)returns the number of rows excluding rows containing duplicates and NULL values in the INV_AMT column.
E. A SELECT statement using the COUNT function with a DISTINCT keyword cannot have a WHERE clause.

Answer: BD
Section: (none)

Explanation/Reference:
Using the COUNT Function
The COUNT function has three formats:
 COUNT(*)
 COUNT(**expr**)
 COUNT(DISTINCT **expr**)
COUNT(*) returns the number of rows in a table that satisfy the criteria of the SELECT statement, including duplicate rows and rows containing null values in any of the columns. If a WHERE clause is included in the SELECT statement, COUNT(*) returns the number of rows that satisfy the condition in the WHERE clause.
In contrast,
COUNT(**expr)** returns the number of non-null values that are in the column identified by **expr**.
COUNT(DISTINCT **expr**) returns the number of unique, non-null values that are in the column identified by **expr**.

QUESTION 109
Examine the structure of the MARKS table:

```
name              Null           Type
---------------------------------------------------------
STUDENT_ID    NOT NULL    VARCHAR2(4)
STUDENT_NAME                  VARCHAR2(25)
SUBJECT1                         NUMBER(3)
SUBJECT2                         NUMBER(3)
SUBJECT3                         NUMBER(3)
```

Which two statements would execute successfully? (Choose two.)

A. SELECT student_name,subject1
FROM marks
WHERE subject1 > AVG(subject1);
B. SELECT student_name,SUM(subject1)
FROM marks
WHERE student_name LIKE 'R%';
C. SELECT SUM(subject1+subject2+subject3)
FROM marks
WHERE student_name IS NULL;
D. SELECT SUM(DISTINCT NVL(subject1,0)), MAX(subject1)
FROM marks
WHERE subject1 > subject2;

Answer: CD
Section: (none)

Explanation/Reference:

QUESTION 110
View the Exhibit and examine the structure of the CUSTOMERS table.

Using the CUSTOMERS table, you need to generate a report that shows the average credit limit for customers in WASHINGTON and NEW YORK.

Which SQL statement would produce the required result?

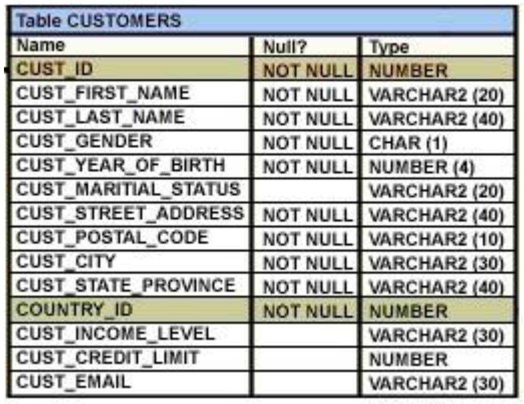


Table CUSTOMERS

Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

A. SELECT cust_city, AVG(cust_credit_limit)
FROM customers
WHERE cust_city IN ('WASHINGTON','NEW YORK')
GROUP BY cust_credit_limit, cust_city;

B. SELECT cust_city, AVG(cust_credit_limit)
FROM customers
WHERE cust_city IN ('WASHINGTON','NEW YORK')
GROUP BY cust_city,cust_credit_limit;

C. SELECT cust_city, AVG(cust_credit_limit)
FROM customers
WHERE cust_city IN ('WASHINGTON','NEW YORK')
GROUP BY cust_city;

D. SELECT cust_city, AVG(NVL(cust_credit_limit,0))
FROM customers
WHERE cust_city IN ('WASHINGTON','NEW YORK');

Answer: C
Section: (none)

Explanation/Reference:
Creating Groups of Data: GROUP BY Clause Syntax
You can use the GROUP BY clause to divide the rows in a table into groups. You can then use the group functions to return summary information for each group.
In the syntax:
group_by_expression Specifies the columns whose values determine the basis for grouping rows
Guidelines
• If you include a group function in a SELECT clause, you cannot select individual results as well, **unless** the individual column appears in the GROUP BY clause. You receive an error message if you fail to include the column list in the GROUP BY clause.
• Using a WHERE clause, you can exclude rows before dividing them into groups.
• You must include the **columns** in the GROUP BY clause.
• You cannot use a column alias in the GROUP BY clause.

QUESTION 111
View the Exhibit and examine the structure of the CUSTOMERS table.

Which statement would display the highest credit limit available in each income level in each city in the

CUSTOMERS table?

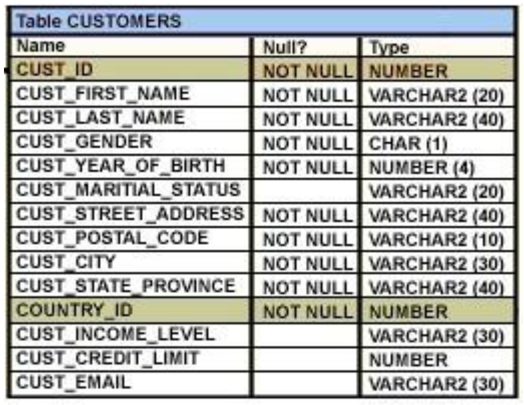


Table CUSTOMERS		
Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

A. SELECT cust_city, cust_income_level, MAX(cust_credit_limit) FROM customers
   GROUP BY cust_city, cust_income_level, cust_credit_limit;
B. SELECT cust_city, cust_income_level, MAX(cust_credit_limit) FROM customers
   GROUP BY cust_city, cust_income_level;
C. SELECT cust_city, cust_income_level, MAX(cust_credit_limit) FROM customers
   GROUP BY cust_credit_limit, cust_income_level, cust_city ;
D. SELECT cust_city, cust_income_level, MAX(cust_credit_limit) FROM customers
   GROUP BY cust_city, cust_income_level, MAX(cust_credit_limit);

Answer: B
Section: (none)

Explanation/Reference:

QUESTION 112
View the Exhibit and examine the structure of the PROMOTIONS table.

Evaluate the following SQL statement:

```
SQL>SELECT promo_category, AVG(promo_cost) Avg_Cost, AVG(promo_cost)*.25 Avg_Overhead
FROM promotions
WHERE UPPER(promo_category) IN ('TV', 'INTERNET','POST')
GROUP BY Avg_Cost
ORDER BY Avg_Overhead;
```

The above query generates an error on execution.

Which clause in the above SQL statement causes the error?

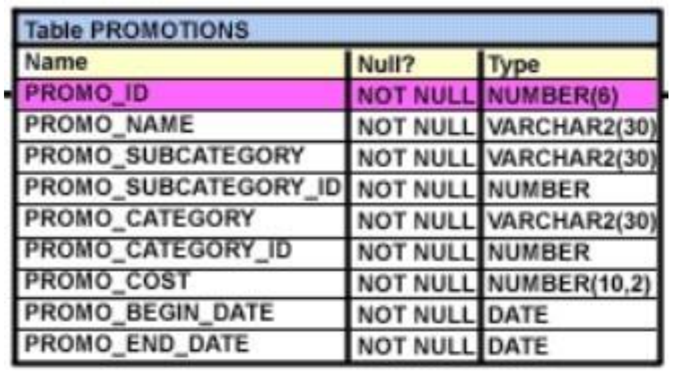


Table PROMOTIONS		
Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

A. WHERE
B. SELECT
C. GROUP BY
D. ORDER BY

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 113
Examine the structure of the ORDERS table:

```
Name Null Type
---------------------------------------------------
ORDER_ID NOT NULL NUMBER(12)
ORDER_DATE NOT NULL TIMESTAMP(6)
CUSTOMER_ID NOT NULL NUMBER(6)
ORDER_STATUS NUMBER(2)
ORDER_TOTAL NUMBER(8,2)
```

You want to find the total value of all the orders for each year and issue the following command:

```
SQL>SELECT TO_CHAR(order_date,'rr'), SUM(order_total)
FROM orders
GROUP BY TO_CHAR(order_date,'yyyy');
```

Which statement is true regarding the outcome?

A. It executes successfully and gives the correct output.
B. It gives an error because the TO_CHAR function is not valid.
C. It executes successfully but does not give the correct output.
D. It gives an error because the data type conversion in the SELECT list does not match the data type conversion in the GROUP BY clause.

Answer: D
Section: (none)

Explanation/Reference:

QUESTION 114
View the Exhibit and examine the structure of the SALES table.

The following query is written to retrieve all those product ID s from the SALES table that have more than 55000 sold and have been ordered more than 10 times.

```
SQL> SELECT prod_id
FROM sales
WHERE quantity_sold > 55000 AND COUNT(*)>10
GROUP BY prod_id
HAVING COUNT(*)>10;
```

Which statement is true regarding this SQL statement?

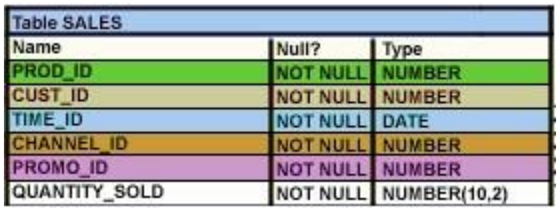


Table SALES		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
CUST_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
CHANNEL_ID	NOT NULL	NUMBER
PROMO_ID	NOT NULL	NUMBER
QUANTITY_SOLD	NOT NULL	NUMBER(10,2)

A. It executes successfully and generates the required result.
B. It produces an error because COUNT(*) should be specified in the SELECT clause also.
C. It produces an error because COUNT(*) should be only in the HAVING clause and not in the WHERE clause.
D. It executes successfully but produces no result because COUNT(prod_id) should be used instead of COUNT(*).

Answer: C
Section: (none)

Explanation/Reference:
Restricting Group Results with the HAVING Clause
You use the HAVING clause to specify the groups that are to be displayed, thus further restricting the groups on the basis of aggregate information.
In the syntax, **group_condition** restricts the groups of rows returned to those groups for which the specified condition is true.
The Oracle server performs the following steps when you use the HAVING clause:
1. Rows are grouped.
2. The group function is applied to the group.
3. The groups that match the criteria in the HAVING clause are displayed.
The HAVING clause can precede the GROUP BY clause, but it is recommended that you place the GROUP BY clause first because it is more logical. Groups are formed and group functions are calculated before the HAVING clause is applied to the groups in the SELECT list.
Note: The WHERE clause restricts rows, whereas the HAVING clause restricts groups.

QUESTION 115
View the Exhibit and examine the structure of the CUSTOMERS table.

Evaluate the following SQL statement:

```
SQL> SELECT cust_city, COUNT(cust_last_name)
FROM customers
WHERE cust_credit_limit > 1000
GROUP BY cust_city
HAVING AVG(cust_credit_limit) BETWEEN 5000 AND 6000;
```

Which statement is true regarding the outcome of the above query?

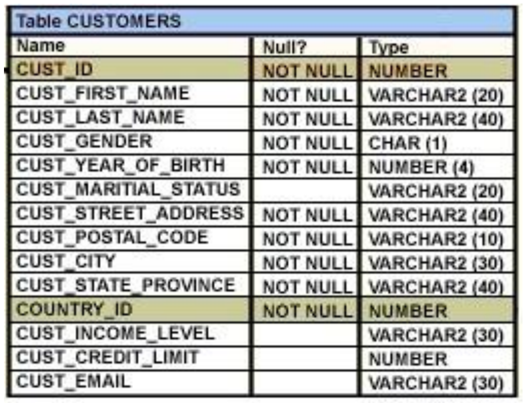


Table CUSTOMERS		
Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

A. It executes successfully.
B. It returns an error because the BETWEEN operator cannot be used in the HAVING clause.
C. It returns an error because WHERE and HAVING clauses cannot be used in the same SELECT statement.
D. It returns an error because WHERE and HAVING clauses cannot be used to apply conditions on the same column.

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 116
You issue the following query:

```
SQL> SELECT AVG(MAX(qty))
FROM ord_items
GROUP BY item_no
HAVING AVG(MAX(qty))>50;
```

Which statement is true regarding the outcome of this query?

A. It executes successfully and gives the correct output.
B. It gives an error because the HAVING clause is not valid.
C. It executes successfully but does not give the correct output.
D. It gives an error because the GROUP BY expression is not valid.

Answer: B
Section: (none)

Explanation/Reference:
The general form of the SELECT statement is further enhanced by the addition of the HAVING clause and becomes:
SELECT column|expression|group_function(column|expression [alias]),…}
FROM table
[WHERE condition(s)]
[GROUP BY {col(s)|expr}]
[HAVING group_condition(s)]
[ORDER BY {col(s)|expr|numeric_pos} [ASC|DESC] [NULLS FIRST|LAST]];
An important difference between the HAVING clause and the other SELECT statement clauses is that it may only be specified if a GROUP BY clause is present. This dependency is sensible since group-level rows must exist before they can be restricted. The HAVING clause can occur before the GROUP BY clause in the SELECT statement. However, it is more common to place the HAVING clause after the GROUP BY clause. All grouping is performed and group functions are executed prior to evaluating the HAVING clause.

QUESTION 117
Which statements are true regarding the WHERE and HAVING clauses in a SELECT statement?

(Choose all that apply.)

A. The HAVING clause can be used with aggregate functions in subqueries.
B. The WHERE clause can be used to exclude rows after dividing them into groups.
C. The WHERE clause can be used to exclude rows before dividing them into groups.
D. The aggregate functions and columns used in the HAVING clause must be specified in the SELECT list of the query.
E. The WHERE and HAVING clauses can be used in the same statement only if they are applied to different columns in the table.

Answer: AC
Section: (none)

Explanation/Reference:

QUESTION 118
View the Exhibit and examine the structure of the PROMOTIONS table.

Examine the following two SQL statements:
Statement 1

```
SQL>SELECT promo_category,SUM(promo_cost)
FROM promotions
WHERE promo_end_date-promo_begin_date > 30
GROUP BY promo_category;
```

Statement 2

```
SQL>SELECT promo_category,sum(promo_cost)
FROM promotions
GROUP BY promo_category
HAVING MIN(promo_end_date-promo_begin_date)>30;
```

Which statement is true regarding the above two SQL statements?

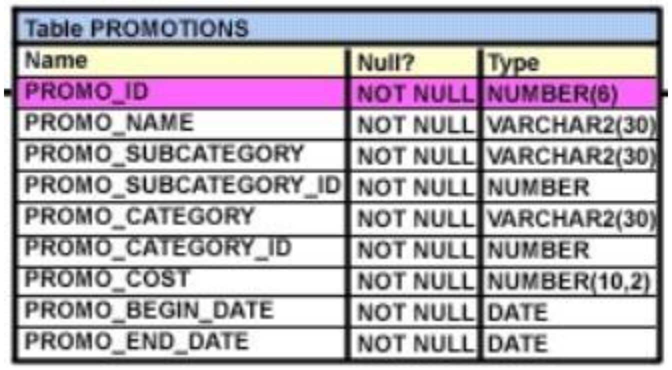


Table PROMOTIONS

Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

A. statement 1 gives an error, statement 2 executes successfully
B. statement 2 gives an error, statement 1 executes successfully
C. statement 1 and statement 2 execute successfully and give the same output
D. statement 1 and statement 2 execute successfully and give a different output

Answer: D
Section: (none)

Explanation/Reference:

QUESTION 119
Examine the data in the ORD_ITEMS table:

```
ORD_NO ITEM_NO QTY
------------------------------------------
1 111 10
1 222 20
1 333 30
2 333 30
2 444 40
3 111 40
```

Evaluate the following query:

```
SQL>SELECT item_no, AVG(qty)
FROM ord_items
HAVING AVG(qty) > MIN(qty) * 2
GROUP BY item_no;
```

Which statement is true regarding the outcome of the above query?

A. It gives an error because the HAVING clause should be specified after the GROUP BY clause.
B. It gives an error because all the aggregate functions used in the HAVING clause must be specified in the SELECT list.
C. It displays the item nos with their average quantity where the average quantity is more than double the minimum quantity of that item in the table.
D. It displays the item nos with their average quantity where the average quantity is more than double the overall minimum quantity of all the items in the table.

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 120
View the Exhibits and examine the structures of the PRODUCTS, SALES, and CUSTOMERS tables.

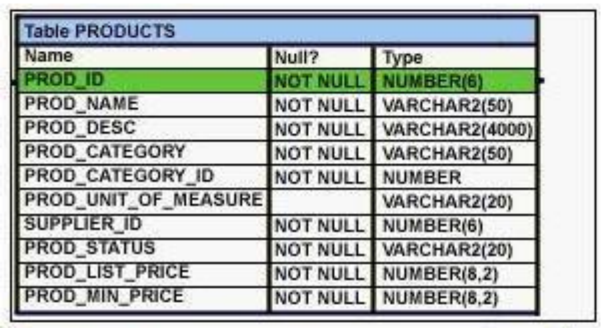


Table PRODUCTS		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

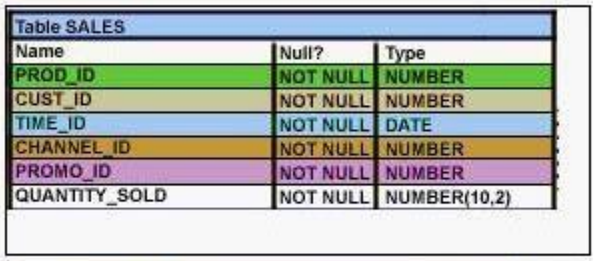


Table SALES		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
CUST_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
CHANNEL_ID	NOT NULL	NUMBER
PROMO_ID	NOT NULL	NUMBER
QUANTITY_SOLD	NOT NULL	NUMBER(10,2)

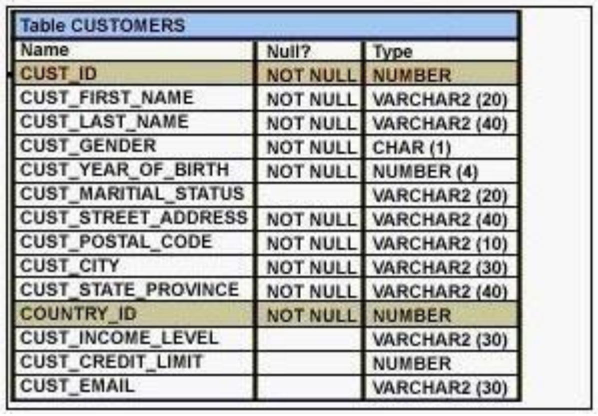


Table CUSTOMERS		
Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

You issue the following query:

```
SQL>SELECT p.prod_id,prod_name,prod_list_price,
quantity_sold,cust_last_name
FROM products p NATURAL JOIN sales s NATURAL JOIN customers c
WHERE prod_id =148;
```

Which statement is true regarding the outcome of this query?

A. It executes successfully.
B. It produces an error because the NATURAL join can be used only with two tables.
C. It produces an error because a column used in the NATURAL join cannot have a qualifier.
D. It produces an error because all columns used in the NATURAL join should have a qualifier.

Answer: C
Section: (none)

Explanation/Reference:
Creating Joins with the USING Clause
Natural joins use all columns with matching names and data types to join the tables. The USING clause can be used to specify only those columns that should be used for an equijoin.
The Natural JOIN USING Clause
The format of the syntax for the natural JOIN USING clause is as follows:
SELECT table1.column, table2.column
FROM table1
JOIN table2 USING (join_column1, join_column2…);
While the pure natural join contains the NATURAL keyword in its syntax, the JOIN…USING syntax does not. An error is raised if the keywords NATURAL and USING occur in the same join clause. The JOIN…USING clause allows one or

more equijoin columns to be explicitly specified in brackets after the USING keyword. This avoids the shortcomings associated with the pure natural join. Many situations demand that tables be joined only on certain columns, and this format caters to this requirement.

QUESTION 121
Which two statements are true regarding the USING clause in table joins? (Choose two .)

A. It can be used to join a maximum of three tables.
B. It can be used to restrict the number of columns used in a NATURAL join.
C. It can be used to access data from tables through equijoins as well as nonequijoins.
D. It can be used to join tables that have columns with the same name and compatible data types.

Answer: BD
Section: (none)

Explanation/Reference:
NATURAL JOIN operation
A NATURAL JOIN is a JOIN operation that creates an implicit join clause for you based on the common columns in the two tables being joined. Common columns are columns that have the same name in both tables.

If the SELECT statement in which the NATURAL JOIN operation appears has an asterisk (*) in the select list, the asterisk will be expanded to the following list of columns (in this order):

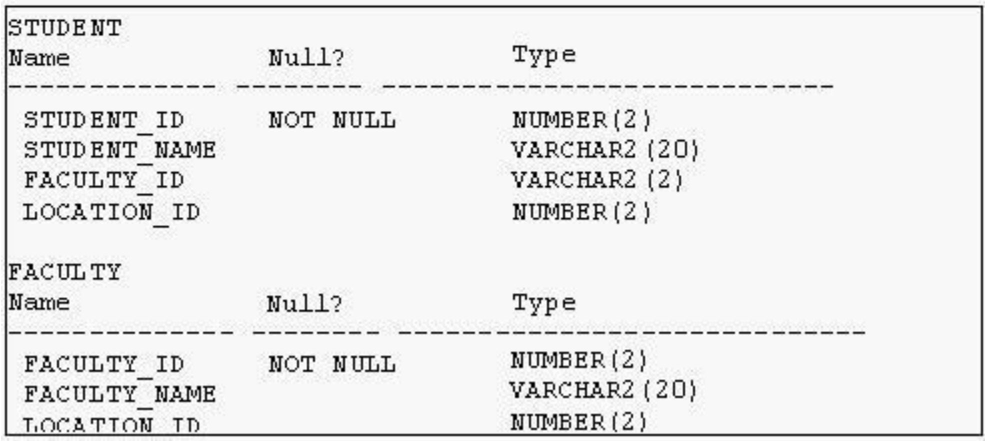
All the common columns
Every column in the first (left) table that is not a common column
Every column in the second (right) table that is not a common column
An asterisk qualified by a table name (for example, COUNTRIES.*) will be expanded to every column of that table that is not a common column.

If a common column is referenced without being qualified by a table name, the column reference points to the column in the first (left) table if the join is an INNER JOIN or a LEFT OUTER JOIN. If it is a RIGHT OUTER JOIN, unqualified references to a common column point to the column in the second (right) table.

Syntax
TableExpression NATURAL [{ LEFT | RIGHT } [OUTER] | INNER] JOIN { TableViewOrFunctionExpression |
(TableExpression) }
Examples
If the tables COUNTRIES and CITIES have two common columns named COUNTRY and COUNTRY_ISO_CODE, the following two SELECT statements are equivalent:

```
SELECT * FROM COUNTRIES NATURAL JOIN CITIES
SELECT * FROM COUNTRIES JOIN CITIES
    USING (COUNTRY, COUNTRY_ISO_CODE)
```

QUESTION 122
View the Exhibit for the structure of the STUDENT and FACULTY tables.



```
STUDENT
Name             Null?          Type
------------- -------- ----------------------------
 STUDENT_ID      NOT NULL       NUMBER(2)
 STUDENT_NAME                   VARCHAR2(20)
 FACULTY_ID                     VARCHAR2(2)
 LOCATION_ID                    NUMBER(2)

FACULTY
Name             Null?          Type
-------------- -------- -----------------------------
 FACULTY_ID      NOT NULL       NUMBER(2)
 FACULTY_NAME                   VARCHAR2(20)
 LOCATION_ID                    NUMBER(2)
```

You need to display the faculty name followed by the number of students handled by the faculty at the base location.

Examine the following two SQL statements:

Statement 1

```
SQL>SELECT faculty_name,COUNT(student_id)
FROM student JOIN faculty
USING (faculty_id, location_id)
GROUP BY faculty_name;
```

Statement 2

```
SQL>SELECT faculty_name,COUNT(student_id)
FROM student NATURAL JOIN faculty
GROUP BY faculty_name;
```

Which statement is true regarding the outcome?

A. Only s tatement 1 executes successfully and gives the required result.
B. Only statement 2 executes successfully and gives the required result.
C. Both statements 1 and 2 execute successfully and give different results.
D. Both statements 1 and 2 execute successfully and give the same required result.

Answer: D
Section: (none)

Explanation/Reference:

QUESTION 123
View the Exhibits and examine the structures of the PRODUCTS, SALES, and CUSTOMERS tables.

You need to generate a report that gives details of the customer's last name, name of the product, and the quantity sold for all customers in 'Tokyo' .

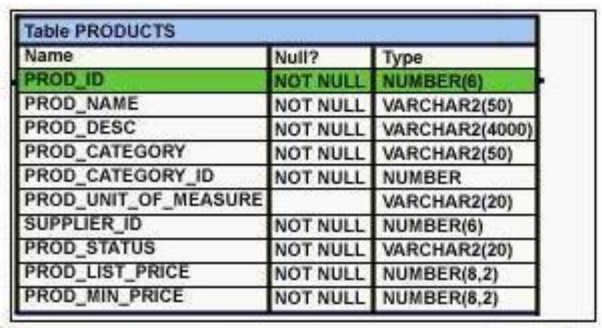


Table PRODUCTS		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

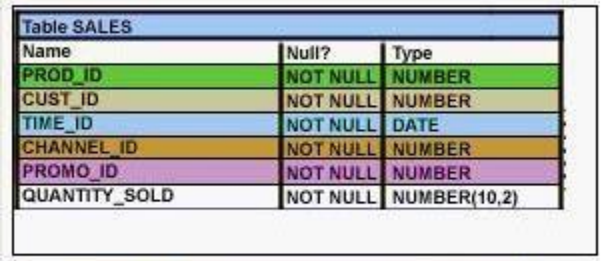


Table SALES		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
CUST_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
CHANNEL_ID	NOT NULL	NUMBER
PROMO_ID	NOT NULL	NUMBER
QUANTITY_SOLD	NOT NULL	NUMBER(10,2)

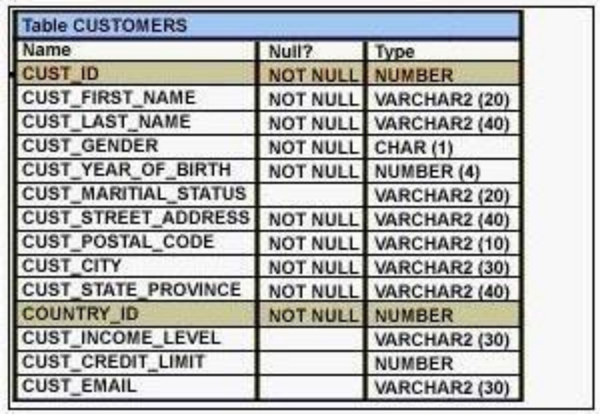


Table CUSTOMERS		
Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

Which two queries give the required result? (Choose two.)

A. SELECT c.cust_last_name,p.prod_name, s.quantity_sold
FROM sales s JOIN products p
USING(prod_id)
JOIN customers c
USING(cust_id)
WHERE c.cust_city='Tokyo';

B. SELECT c.cust_last_name, p.prod_name, s.quantity_sold
FROM products p JOIN sales s JOIN customers c
ON(p.prod_id=s.prod_id)
ON(s.cust_id=c.cust_id)
WHERE c.cust_city='Tokyo';

C. SELECT c.cust_last_name, p.prod_name, s.quantity_sold
FROM products p JOIN sales s
ON(p.prod_id=s.prod_id)
JOIN customers c
ON(s.cust_id=c.cust_id)
AND c.cust_city='Tokyo';

D. SELECT c.cust_id,c.cust_last_name,p.prod_id, p.prod_name, s.quantity_sold FROM products p JOIN sales s
USING(prod_id)
JOIN customers c
USING(cust_id)
WHERE c.cust_city='Tokyo';

Answer: AC
Section: (none)

Explanation/Reference:

QUESTION 124
View the Exhibit and examine the structure of the PROMOTIONS, SALES, and CUSTOMER tables.

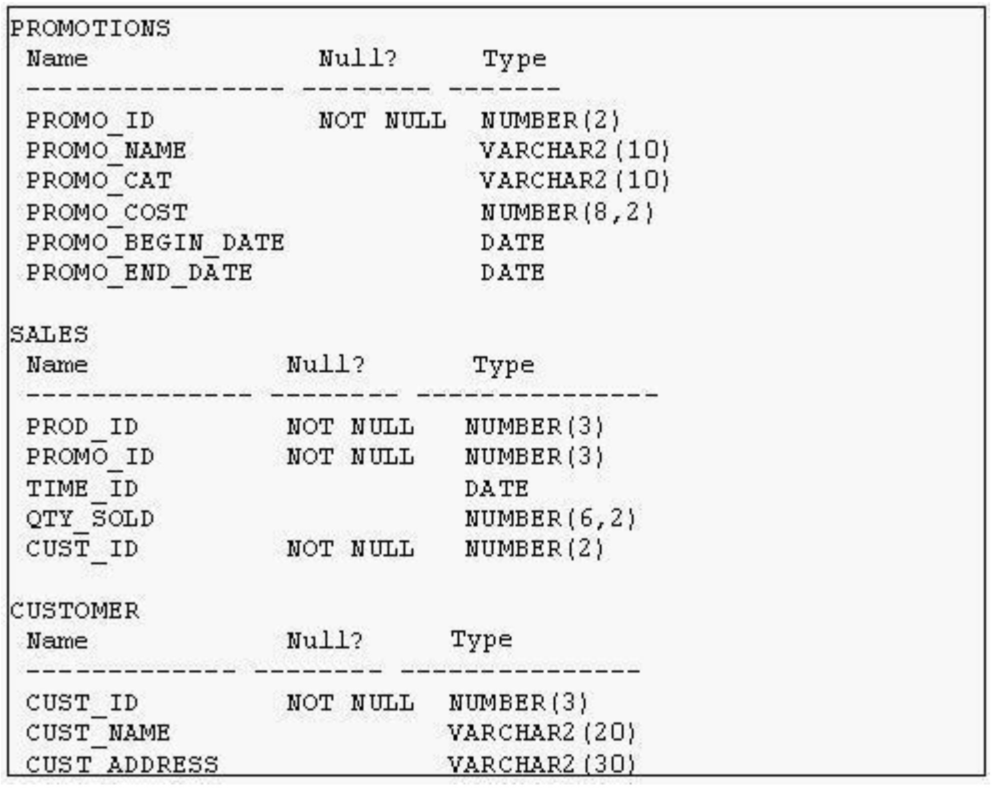
You need to generate a report showing the promo name along with the customer name for all products that were sold during their promo campaign and before 30th October 2007.

You issue the following query:

```
SQL> SELECT promo_name,cust_name
FROM promotions p JOIN sales s
ON(time_id BETWEEN promo_begin_date AND promo_end_date)
JOIN customer c
ON (s.cust_id = c.cust_id) AND time_id < '30-oct-2007';
```

Which statement is true regarding the above query?

Exhibit:



```
PROMOTIONS
 Name                Null?     Type
 ---------------- -------- -------
 PROMO_ID            NOT NULL  NUMBER(2)
 PROMO_NAME                    VARCHAR2(10)
 PROMO_CAT                     VARCHAR2(10)
 PROMO_COST                    NUMBER(8,2)
 PROMO_BEGIN_DATE              DATE
 PROMO_END_DATE                DATE

SALES
 Name              Null?      Type
 -------------- -------- ---------------
 PROD_ID           NOT NULL   NUMBER(3)
 PROMO_ID          NOT NULL   NUMBER(3)
 TIME_ID                      DATE
 QTY_SOLD                     NUMBER(6,2)
 CUST_ID           NOT NULL   NUMBER(2)

CUSTOMER
 Name              Null?     Type
 ------------- -------- ---------------
 CUST_ID           NOT NULL  NUMBER(3)
 CUST_NAME                   VARCHAR2(20)
 CUST_ADDRESS                VARCHAR2(30)
```

A. It executes successfully and gives the required result.
B. It executes successfully but does not give the required result.
C. It produces an error because the join order of the tables is incorrect.
D. It produces an error because equijoin and nonequijoin conditions cannot be used in the same SELECT statement.

Answer: B
Section: (none)

Explanation/Reference:

QUESTION 125
Examine the structure of the CUSTOMERS table:

name Null Type
CUSTNO NOT NULL NUMBER(3)
CUSTNAME NOT NULL VARCHAR2(25)
CUSTADDRESS VARCHAR2(35)
CUST_CREDIT_LIMIT NUMBER(5)

CUSTNO is the PRIMARY KEY in the table. You want to find out if any customers' details have been entered more than once using different CUSTNO, by listing all the duplicate names.

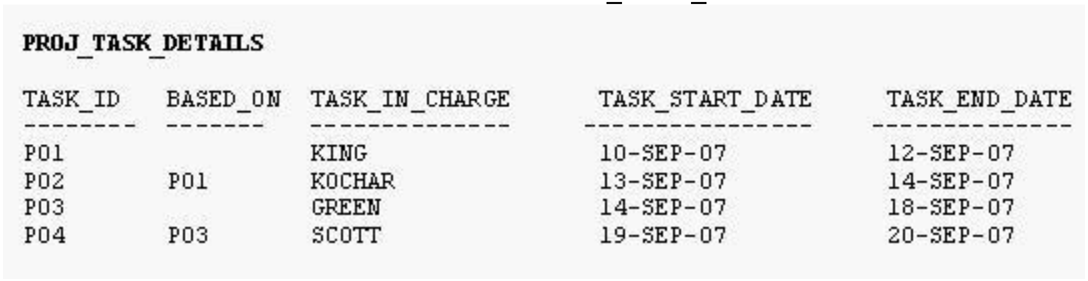
Which two methods can you use to get the required result? (Choose two.)

A. self-join
B. subquery
C. full outer-join with self-join
D. left outer-join with self-join
E. right outer-join with self-join

Answer: AB
Section: (none)

Explanation/Reference:

QUESTION 126
View the Exhibit and examine the data in the PROJ_TASK_DETAILS table.



```
PROJ_TASK_DETAILS

TASK_ID   BASED_ON  TASK_IN_CHARGE    TASK_START_DATE    TASK_END_DATE
--------  -------   --------------  ----------------   --------------
P01                 KING              10-SEP-07          12-SEP-07
P02       P01       KOCHAR            13-SEP-07          14-SEP-07
P03                 GREEN             14-SEP-07          18-SEP-07
P04       P03       SCOTT             19-SEP-07          20-SEP-07
```

The PROJ_TASK_DETAILS table stores information about tasks involved in a project and the relation between them. The BASED_ON column indicates dependencies between tasks. Some tasks do not depend on the completion of any other tasks.
You need to generate a report showing all task IDs, the corresponding task ID they are dependent on, and the name of the employee in charge of the task it depends on.

Which query would give the required result?

A. SELECT p.task_id, p.based_on, d.task_in_charge
FROM proj_task_details p JOIN proj_task_details d
ON (p.based_on = d.task_id);
B. SELECT p.task_id, p.based_on, d.task_in_charge
FROM proj_task_details p LEFT OUTER JOIN proj_task_details d ON (p.based_on = d.task_id);
C. SELECT p.task_id, p.based_on, d.task_in_charge
FROM proj_task_details p FULL OUTER JOIN proj_task_details d ON (p.based_on = d.task_id);
D. SELECT p.task_id, p.based_on, d.task_in_charge
FROM proj_task_details p JOIN proj_task_details d
ON (p.task_id = d.task_id);

Answer: B
Section: (none)

Explanation/Reference:

QUESTION 127
Examine the data in the CUSTOMERS table:

```
CUSTNO     CUSTNAME CITY
------------------------------------------------
1                    KING              SEATTLE
2                    GREEN          BOSTON
3                    KOCHAR       SEATTLE
4                    SMITH             NEW YORK
```

You want to list all cities that have more than one customer along with the customer details.
Evaluate the following query:

```
SQL>SELECT c1.custname, c1.city
FROM Customers c1 __________________ Customers c2
ON (c1.city=c2.city AND c1.custname<>c2.custname);
```

Which two JOIN options can be used in the blank in the above query to give the correct output? (Choose two.)

A. JOIN
B. NATURAL JOIN
C. LEFT OUTER JOIN
D. FULL OUTER JOIN
E. RIGHT OUTER JOIN

Answer: AE
Section: (none)

Explanation/Reference:

QUESTION 128
View the Exhibits and examine the structures of the CUSTOMERS, SALES, and COUNTRIES tables.

You need to generate a report that shows all country names, with corresponding customers (if any) and sales details (if any), for all customers.

Which FROM clause gives the required result?

A. FROM sales JOIN customers USING (cust_id)
   FULL OUTER JOIN countries USING (country_id);
B. FROM sales JOIN customers USING (cust_id)
   RIGHT OUTER JOIN countries USING (country_id);
C. FROM customers LEFT OUTER JOIN sales USING (cust_id)
   RIGHT OUTER JOIN countries USING (country_id);
D. FROM customers LEFT OUTER JOIN sales USING (cust_id)
   LEFT OUTER JOIN countries USING (country_id);

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 129
View the Exhibits and examine the structures of the PROMOTIONS and SALES tables.

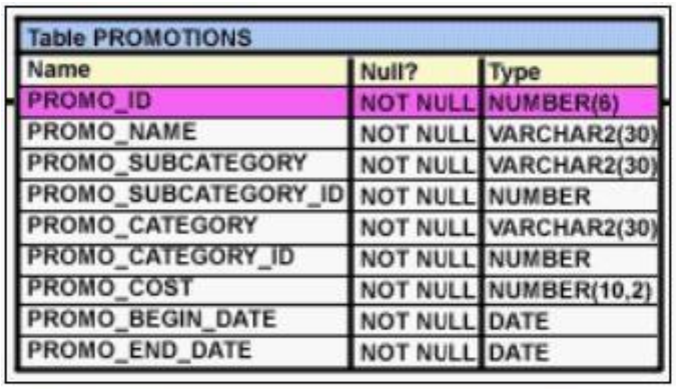


Table PROMOTIONS		
Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

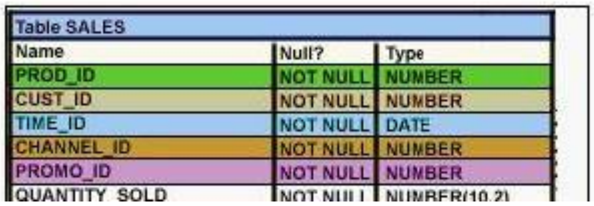


Table SALES		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
CUST_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
CHANNEL_ID	NOT NULL	NUMBER
PROMO_ID	NOT NULL	NUMBER
QUANTITY_SOLD	NOT NULL	NUMBER(10,2)

Evaluate the following SQL statement:

```
SQL>SELECT p.promo_id, p.promo_name, s.prod_id
FROM sales s RIGHT OUTER JOIN promotions p
ON (s.promo_id = p.promo_id);
```

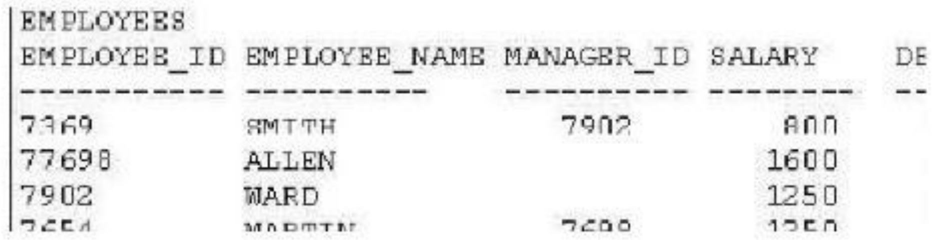
Which statement is true regarding the output of the above query?

A. It gives the details of promos for which there have been sales.
B. It gives the details of promos for which there have been no sales.
C. It gives details of all promos irrespective of whether they have resulted in a sale or not.
D. It gives details of product ID s that have been sold irrespective of whether they had a promo or not.

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 130
View the Exhibit and examine the data in the EMPLOYEES table:



```
EMPLOYEES
EMPLOYEE_ID EMPLOYEE_NAME MANAGER_ID SALARY    DE
----------- ----------    ---------- --------  --
7369        SMITH              7902       800
77698       ALLEN                        1600
7902        WARD                         1250
7654        MARTIN             7698      1250
```

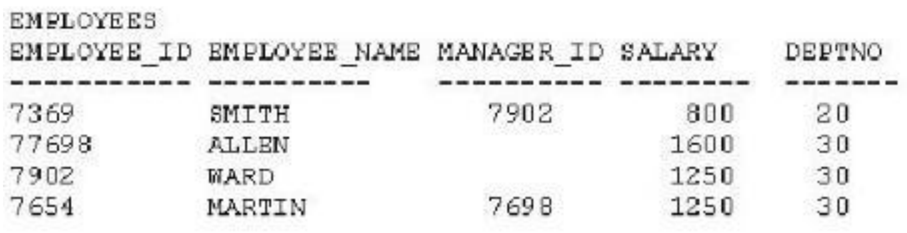
You want to display all the employee names and their corresponding manager names.

Evaluate the following query:

```
SQL> SELECT e.employee_name "EMP NAME", m.employee_name "MGR NAME"
FROM employees e ______________ employees m
ON e.manager_id = m.employee_id;
```

Which JOIN option can be used in the blank in the above query to get the required output?

Exhibit:



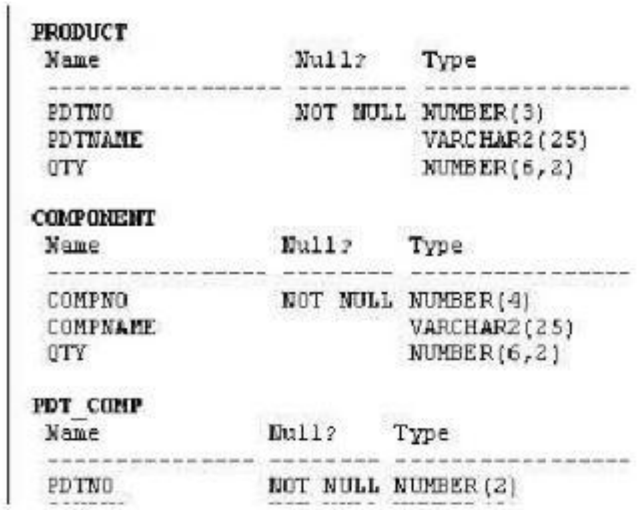
```
EMPLOYEES
EMPLOYEE_ID EMPLOYEE_NAME MANAGER_ID SALARY    DEPTNO
----------- ----------    ---------- --------  -------
7369        SMITH            7902        800     20
77698       ALLEN                       1600     30
7902        WARD                        1250     30
7654        MARTIN           7698       1250     30
```

A. o nly inner JOIN
B. only FULL OUTER JOIN
C. only LEFT OUTER JOIN
D. only RIGHT OUTER JOIN

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 131
View the Exhibit and examine the structure of the PRODUCT, COMPONENT, and PDT_COMP tables.



```
PRODUCT
 Name               Null?    Type
 ------------------ -------- ---------------
 PDTNO              NOT NULL NUMBER(3)
 PDTNAME                     VARCHAR2(25)
 QTY                         NUMBER(6,2)

COMPONENT
 Name              Null?    Type
 ----------------- -------- ---------------
 COMPNO            NOT NULL NUMBER(4)
 COMPNAME                   VARCHAR2(25)
 QTY                        NUMBER(6,2)

PDT_COMP
 Name             Null?    Type
 ---------------- -------- ----------------
 PDTNO            NOT NULL NUMBER(2)
```

In PRODUCT table, PDTNO is the primary key.
In COMPONENT table, COMPNO is the primary key.
In PDT_COMP table, (PDTNO,COMPNO) is the primary key, PDTNO is the foreign key referencing
PDTNO in PRODUCT table and COMPNO is the foreign key referencing the COMPNO in COMPONENT table.

You want to generate a report listing the product names and their corresponding component names, if the component names and product names exist.

Evaluate the following query:

SQL>SELECT pdtno,pdtname, compno,compname
FROM product _____________ pdt_comp
USING (pdtno) ____________ component USING(compno)
WHERE compname IS NOT NULL;

Which combination of joins used in the blanks in the above query gives the correct output?

A. JOIN; JOIN
B. FULL OUTER JOIN; FULL OUTER JOIN

C. RIGHT OUTER JOIN; LEFT OUTER JOIN
D. LEFT OUTER JOIN; RIGHT OUTER JOIN

Answer: C
Section: (none)

Explanation/Reference:

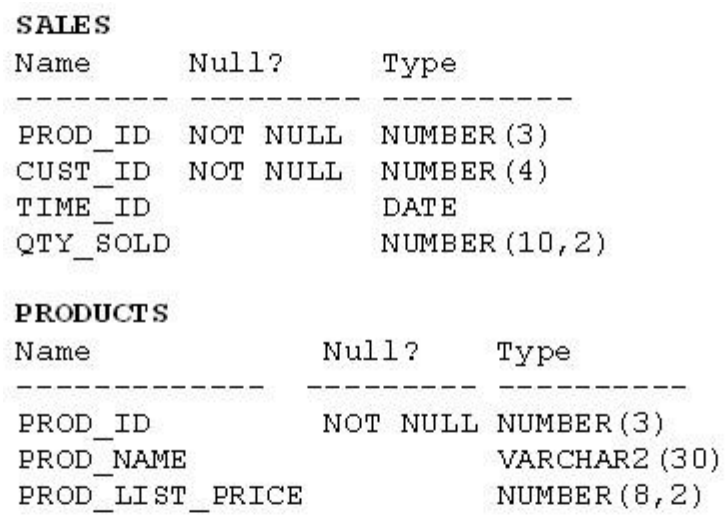
QUESTION 132
View the Exhibit and examine the structure of the SALES and PRODUCTS tables.

In the SALES table, PROD_ID is the foreign key referencing PROD_ID in the PRODUCTS table,
You want to list each product ID and the number of times it has been sold.

Evaluate the following query:

```
SQL>SELECT p.prod_id, COUNT(s.prod_id)
FROM products p _____________ sales s
ON p.prod_id = s.prod_id
GROUP BY p.prod_id;
```

Which two JOIN options can be used in the blank in the above query to get the required output? (Choose two.)



```
SALES
Name     Null?     Type
-------- --------- ----------
PROD_ID  NOT NULL  NUMBER(3)
CUST_ID  NOT NULL  NUMBER(4)
TIME_ID            DATE
QTY_SOLD           NUMBER(10,2)

PRODUCTS
Name                Null?     Type
-------------       --------- ----------
PROD_ID             NOT NULL  NUMBER(3)
PROD_NAME                     VARCHAR2(30)
PROD_LIST_PRICE               NUMBER(8,2)
```

A. JOIN
B. FULL OUTER JOIN
C. LEFT OUTER JOIN
D. RIGHT OUTER JOIN

Answer: BC
Section: (none)

Explanation/Reference:

QUESTION 133
Which two statements are true regarding subqueries? (Choose two.)

A. A subquery can retrieve zero or more rows.
B. Only two subqueries can be placed at one level.

C. A subquery can be used only in SQL query statements.
D. A subquery can appear on either side of a comparison operator.
E. There is no limit on the number of subquery levels in the WHERE clause of a SELECT statement.

Answer: AD
Section: (none)

Explanation/Reference:

Using a Subquery to Solve a Problem

Suppose you want to write a query to find out who earns a salary greater than Abel's salary. To solve this problem, you need **two** queries: one to find how much Abel earns, and a second query to find who earns more than that amount.
You can solve this problem by combining the two queries, placing one query **inside** the other query.
The inner query (or **subquery**) returns a value that is used by the outer query (or **main query**).
Using a subquery is equivalent to performing two sequential queries and using the result of the first query as the search value in the second query.

Subquery Syntax

A subquery is a SELECT statement that is embedded in the clause of another SELECT statement.
You can build powerful statements out of simple ones by using subqueries. They can be very useful when you need to select rows from a table with a condition that depends on the data in the table itself.
You can place the subquery in a number of SQL clauses, including the following:
 WHERE clause
 HAVING clause
 FROM clause
In the syntax:
operator includes a comparison condition such as >, =, or IN
Note: Comparison conditions fall into two classes: single-row operators (>, =, >=, <, <>, <=) and multiple-row operators (IN, ANY, ALL, EXISTS).
The subquery is often referred to as a nested SELECT, sub-SELECT, or inner SELECT statement.
The subquery generally executes first, and its output is used to complete the query condition for the main (or outer) query.

Guidelines for Using Subqueries

 Enclose subqueries in parentheses. Place subqueries on the right side of the comparison condition for readability. (However, the subquery can appear on either side of the comparison operator.) Use single-row operators with single-row subqueries and multiple-row operators with multiple-row subqueries.

Subqueries can be nested to an unlimited depth in a FROM clause but to "only" 255 levels in a WHERE clause. They can be used in the SELECT list and in the FROM, WHERE, and HAVING clauses of a query.

QUESTION 134
Where can subqueries be used? (Choose all that apply.)

A. field names in the SELECT statement
B. the FROM clause in the SELECT statement
C. the HAVING clause in the SELECT statement
D. the GROUP BY clause in the SELECT statement
E. the WHERE clause in only the SELECT statement
F. the WHERE clause in SELECT as well as all DML statements

Answer: ABCF

Section: (none)

Explanation/Reference:
SUBQUERIES can be used in the SELECT list and in the FROM, WHERE, and HAVING clauses of a query.

A subquery can have any of the usual clauses for selection and projection. The following are required clauses:
■ A SELECT list
■ A FROM clause
The following are optional clauses:
■ WHERE
■ GROUP BY
■ HAVING
The subquery (or subqueries) within a statement must be executed before the parent query that calls it, in order that the results of the subquery can be passed to the parent.

QUESTION 135
Which three statements are true regarding subqueries? (Choose three.)

A. Subqueries can contain GROUP BY and ORDER BY clauses.
B. Main query and subquery can get data from different tables.
C. Main query and subquery must get data from the same tables.
D. Subqueries can contain ORDER BY but not the GROUP BY clause.
E. Only one column or expression can be compared between the main query and subquery.
F. Multiple columns or expressions can be compared between the main query and subquery.

Answer: ABF
Section: (none)

Explanation/Reference:
SUBQUERIES can be used in the SELECT list and in the FROM, WHERE, and HAVING clauses of a query.

A subquery can have any of the usual clauses for selection and projection. The following are required clauses:
■ A SELECT list
■ A FROM clause
The following are optional clauses:
■ WHERE
■ GROUP BY
■ HAVING
The subquery (or subqueries) within a statement must be executed before the parent query that calls it, in order that the results of the subquery can be passed to the parent.

QUESTION 136
View the Exhibit and examine the structure of the PRODUCTS table.

Which two tasks would require subqueries? (Choose two.)

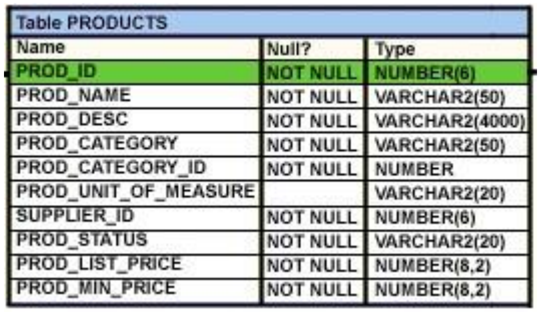


Table PRODUCTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

A. Display the minimum list price for each product status.
B. Display all suppliers whose list price is less than 1000.
C. Display the number of products whose list price is more than the average list price.
D. Display the total number of products supplied by supplier 102 and have product status as 'obsolete'.
E. Display all products whose minimum list price is more than the average list price of products and have the status 'orderable'.

Answer: CE
Section: (none)

Explanation/Reference:

QUESTION 137
View the Exhibits and examine PRODUCTS and SALES tables.

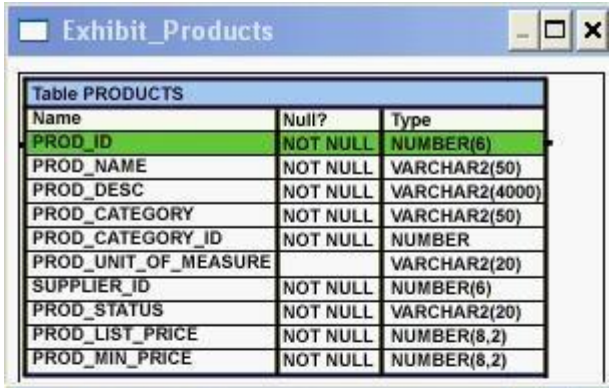

Exhibit_Products

Table PRODUCTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

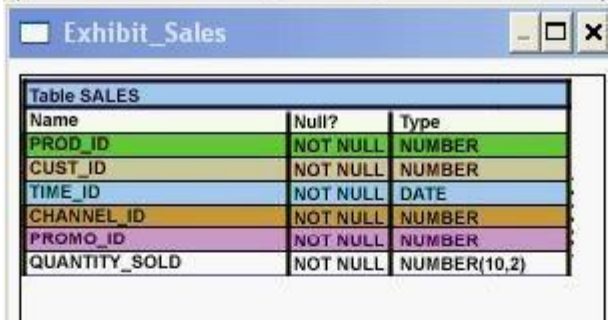

Exhibit_Sales

Table SALES

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
CUST_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
CHANNEL_ID	NOT NULL	NUMBER
PROMO_ID	NOT NULL	NUMBER
QUANTITY_SOLD	NOT NULL	NUMBER(10,2)

You issue the following query to display product name and the number of times the product has been sold:

```
SQL>SELECT p.prod_name, i.item_cnt
FROM (SELECT prod_id, COUNT(*) item_cnt
FROM sales
GROUP BY prod_id) i RIGHT OUTER JOIN products p
ON i.prod_id = p.prod_id;
```

What happens when the above statement is executed?

A. The statement executes successfully and produces the required output.
B. The statement produces an error because ITEM_CNT cannot be displayed in the outer query.
C. The statement produces an error because a subquery in the FROM clause and outer-joins cannot be used together.
D. The statement produces an error because the GROUP BY clause cannot be used in a subquery in the FROM clause.

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 138
Which statement is true regarding subqueries?

A. The LIKE operator cannot be used with single- row subqueries.
B. The NOT IN operator is equivalent to IS NULL with single- row subqueries.
C. =ANY and =ALL operators have the same functionality in multiple- row subqueries.
D. The NOT operator can be used with IN, ANY, and ALL operators in multiple- row subqueries.

Answer: D
Section: (none)

Explanation/Reference:
Using the ANY Operator in Multiple-Row Subqueries
The ANY operator (and its synonym, the SOME operator) compares a value to **each** value returned by a subquery.
<ANY means less than the maximum.
>ANY means more than the minimum.
=ANY is equivalent to IN
Using the ALL Operator in Multiple-Row Subqueries
The ALL operator compares a value to **every** value returned by a subquery.
>ALL means more than the maximum and
<ALL means less than the minimum.
The NOT operator can be used with IN, ANY, and ALL operators.

QUESTION 139
Which three statements are true about multiple-row subqueries? (Choose three.)

A. They can contain a subquery within a subquery.
B. They can return multiple columns as well as rows.
C. They cannot contain a subquery within a subquery.
D. They can return only one column but multiple rows.
E. They can contain group functions and GROUP BY and HAVING clauses.
F. They can contain group functions and the GROUP BY clause, but not the HAVING clause.

Answer: ABE
Section: (none)

Explanation/Reference:

QUESTION 140
Examine the structure of the PRODUCTS table:

name Null Type
PROD_ID NOT NULL NUMBER(4)
PROD_NAME VARCHAR2(20)
PROD_STATUS VARCHAR2(6)
QTY_IN_HAND NUMBER(8,2)
UNIT_PRICE NUMBER(10,2)

You want to display the names of the products that have the highest total value for UNIT_PRICE * QTY_IN_HAND.

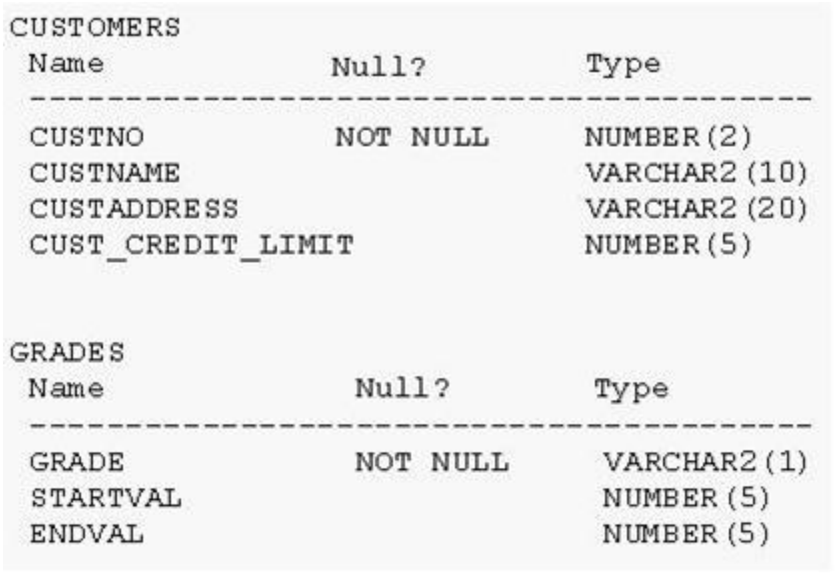
Which SQL statement gives the required output?

A. SELECT prod_name
FROM products
WHERE (unit_price * qty_in_hand) = (SELECT MAX(unit_price * qty_in_hand) FROM products);
B. SELECT prod_name
FROM products
WHERE (unit_price * qty_in_hand) = (SELECT MAX(unit_price * qty_in_hand) FROM products
GROUP BY prod_name);
C. SELECT prod_name
FROM products
GROUP BY prod_name
HAVING MAX(unit_price * qty_in_hand) = (SELECT MAX(unit_price * qty_in_hand) FROM products
GROUP BY prod_name);
D. SELECT prod_name
FROM products
WHERE (unit_price * qty_in_hand) = (SELECT MAX(SUM(unit_price * qty_in_hand)) FROM products)
GROUP BY prod_name;

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 141
View the Exhibit and examine the structure of CUSTOMERS and GRADES tables.



```
CUSTOMERS
 Name               Null?        Type
 -----------------------------------------
 CUSTNO             NOT NULL     NUMBER(2)
 CUSTNAME                        VARCHAR2(10)
 CUSTADDRESS                     VARCHAR2(20)
 CUST_CREDIT_LIMIT               NUMBER(5)


GRADES
 Name                 Null?        Type
 -----------------------------------------
 GRADE                NOT NULL     VARCHAR2(1)
 STARTVAL                          NUMBER(5)
 ENDVAL                            NUMBER(5)
```

You need to display names and grades of customers who have the highest credit limit.

Which two SQL statements would accomplish the task? (Choose two.)

A. SELECT custname, grade
FROM customers, grades
WHERE (SELECT MAX(cust_credit_limit)
FROM customers) BETWEEN startval and endval;
B. SELECT custname, grade
FROM customers, grades
WHERE (SELECT MAX(cust_credit_limit)
FROM customers) BETWEEN startval and endval
AND cust_credit_limit BETWEEN startval AND endval;
C. SELECT custname, grade
FROM customers, grades

WHERE cust_credit_limit = (SELECT MAX(cust_credit_limit)
FROM customers)
AND cust_credit_limit BETWEEN startval AND endval;

D. SELECT custname, grade
FROM customers , grades
WHERE cust_credit_limit IN (SELECT MAX(cust_credit_limit)
FROM customers)
AND MAX(cust_credit_limit) BETWEEN startval AND endval;

Answer: BC
Section: (none)

Explanation/Reference:

QUESTION 142
View the Exhibit and examine the structure of the PRODUCTS table.

Evaluate the following query:

```
SQL> SELECT prod_name
FROM products
WHERE prod_id IN (SELECT prod_id FROM products
WHERE prod_list_price =
(SELECT MAX(prod_list_price)FROM products
WHERE prod_list_price <
(SELECT MAX(prod_list_price)FROM products)));
```

What would be the outcome of executing the above SQL statement?

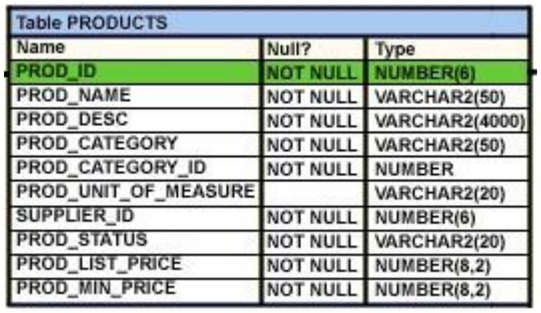


Table PRODUCTS		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

A. It produces an error.
B. It shows the names of all products in the table.
C. It shows the names of products whose list price is the second highest in the table.
D. It shows the names of all products whose list price is less than the maximum list price.

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 143
View the Exhibit and examine the structure of the PROMOTIONS table.

You have to generate a report that displays the promo name and start date for all promos that started after

the last promo in the 'INTERNET' category.

Which query would give you the required output?

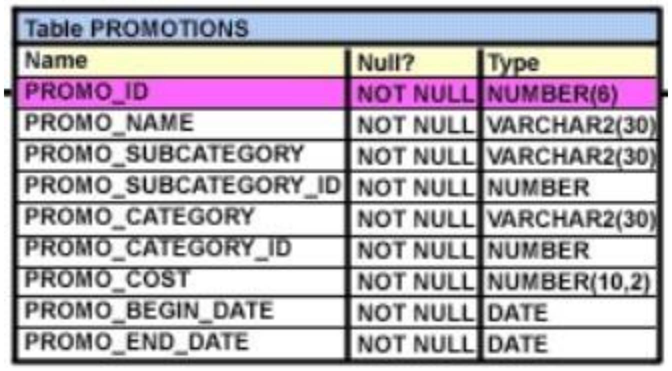


Table PROMOTIONS		
Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

A. SELECT promo_name, promo_begin_date FROM promotions
   WHERE promo_begin_date > ALL (SELECT MAX(promo_begin_date) FROM promotions)AND
   promo_category = 'INTERNET';
B. SELECT promo_name, promo_begin_date FROM promotions
   WHERE promo_begin_date IN (SELECT promo_begin_date
   FROM promotions
   WHERE promo_category='INTERNET');
C. SELECT promo_name, promo_begin_date FROM promotions
   WHERE promo_begin_date > ALL (SELECT promo_begin_date
   FROM promotions
   WHERE promo_category = 'INTERNET');
D. SELECT promo_name, promo_begin_date FROM promotions
   WHERE promo_begin_date > ANY (SELECT promo_begin_date
   FROM promotions
   WHERE promo_category = 'INTERNET');

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 144
View the Exhibit and examine the structure of the PRODUCTS table.

You want to display the category with the maximum number of items.

You issue the following query:

```
SQL>SELECT COUNT(*),prod_category_id
FROM products
GROUP BY prod_category_id
HAVING COUNT(*) = (SELECT MAX(COUNT(*)) FROM products);
```

What is the outcome?

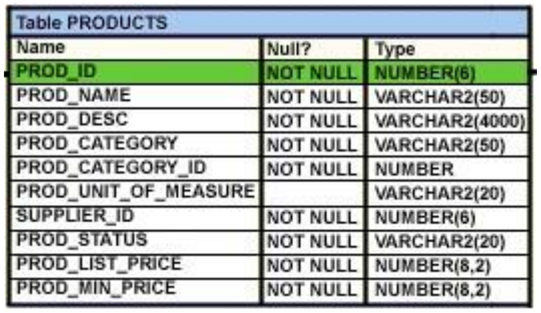


Table PRODUCTS		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

A. It executes successfully and gives the correct output.
B. It executes successfully but does not give the correct output.
C. It generates an error because the subquery does not have a GROUP BY clause.
D. It generates an error because = is not valid and should be replaced by the IN operator.

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 145
View the Exhibit and examine the structure of the CUSTOMERS table.

You issue the following SQL statement on the CUSTOMERS table to display the customers who are in the
same country as customers with the last name 'KING' and whose credit limit is less than the maximum credit limit in countries
that have customers with the last name 'KING':

```
SQL> SELECT cust_id,cust_last_name
FROM customers
WHERE country_id IN(SELECT country_id
FROM customers
WHERE cust_last_name ='King')
AND cust_credit_limit < (SELECT MAX(cust_credit_limit)
FROM customers
WHERE country_id IN(SELECT country_id
FROM customers
WHERE cust_last_name='King'));
```

Which statement is true regarding the outcome of the above query?

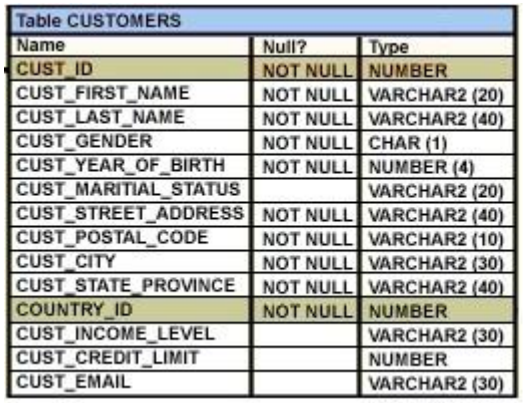


Table CUSTOMERS		
Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

A. It executes and shows the required result.
B. It produces an error and the < operator should be replaced by < ALL to get the required output.

C. It produces an error and the < operator should be replaced by < ANY to get the required output.
D. It produces an error and the IN operator should be replaced by = in the WHERE clause of the main query to get the required output.

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 146
Evaluate the following SQL statement:

```
SQL> SELECT cust_id, cust_last_name
FROM customers
WHERE cust_credit_limit IN
(select cust_credit_limit
FROM customers
WHERE cust_city ='Singapore');
```

Which statement is true regarding the above query if one of the values generated by the subquery is NULL?

A. It produces an error.
B. It executes but returns no rows.
C. It generates output for NULL as well as the other values produced by the subquery.
D. It ignores the NULL value and generates output for the other values produced by the subquery.

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 147
View the Exhibit and examine the structure of the PROMOTIONS table.

Evaluate the following SQL statement:

```
SQL>SELECT promo_name,CASE
WHEN promo_cost >=(SELECT AVG(promo_cost)
FROM promotions
WHERE promo_category='TV')
then 'HIGH'
else 'LOW'
END COST_REMARK
FROM promotions;
```

Which statement is true regarding the outcome of the above query?

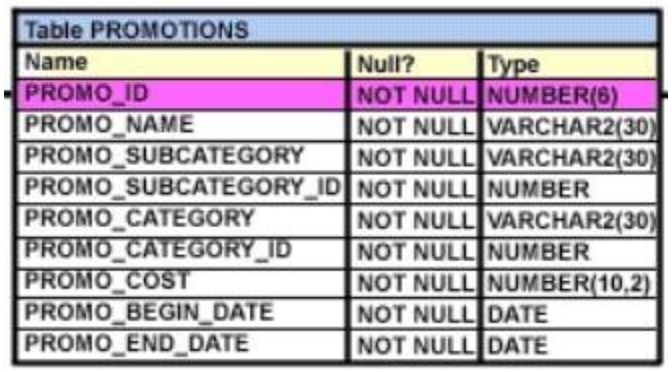


Table PROMOTIONS		
Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

A. It shows COST_REMARK for all the promos in the table.
B. It produces an error because the subquery gives an error.
C. It shows COST_REMARK for all the promos in the promo category 'TV'.
D. It produces an error because subqueries cannot be used with the CASE expression.

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 148
View the Exhibit and examine the structure of the PRODUCTS tables.

You want to generate a report that displays the average list price of product categories where the average list price is less than half the maximum in each category.

Which query would give the correct output?

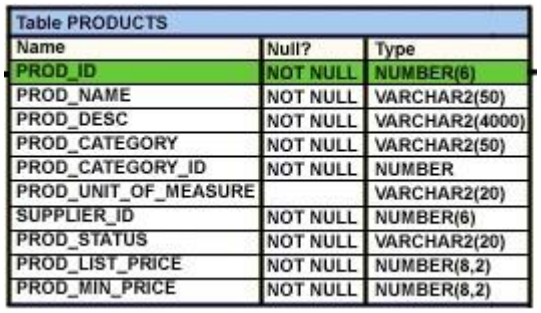


Table PRODUCTS		
Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

A. SELECT prod_category,avg(prod_list_price)
FROM products
GROUP BY prod_category
HAVING avg(prod_list_price) < ALL
(SELECT max(prod_list_price)/2
FROM products
GROUP BY prod_category);
B. SELECT prod_category,avg(prod_list_price)
FROM products
GROUP BY prod_category
HAVING avg(prod_list_price) > ANY
(SELECT max(prod_list_price)/2
FROM products
GROUP BY prod_category);
C. SELECT prod_category,avg(prod_list_price)
FROM products

```
HAVING avg(prod_list_price) < ALL
(SELECT max(prod_list_price)/2
FROM products
GROUP BY prod_category);
```
D. SELECT prod_category,avg(prod_list_price)
```
FROM products
GROUP BY prod_category
HAVING avg(prod_list_price) > ANY
(SELECT max(prod_list_price)/2
FROM products);
```

Answer: A
Section: (none)

Explanation/Reference:
Using the ANY Operator in Multiple-Row Subqueries
The ANY operator (and its synonym, the SOME operator) compares a value to **each** value returned by a subquery.
<ANY means less than the maximum.
>ANY means more than the minimum.
=ANY is equivalent to IN
Using the ALL Operator in Multiple-Row Subqueries
The ALL operator compares a value to **every** value returned by a subquery.
>ALL means more than the maximum and
<ALL means less than the minimum.
The NOT operator can be used with IN, ANY, and ALL operators.

QUESTION 149
View the Exhibits and examine the structures of the COSTS and PROMOTIONS tables.

Evaluate the following SQL statement:

```
SQL> SELECT prod_id FROM costs
WHERE promo_id IN (SELECT promo_id FROM promotions
WHERE promo_cost < ALL
(SELECT MAX(promo_cost) FROM promotions
GROUP BY (promo_end_date-
promo_begin_date)));
```

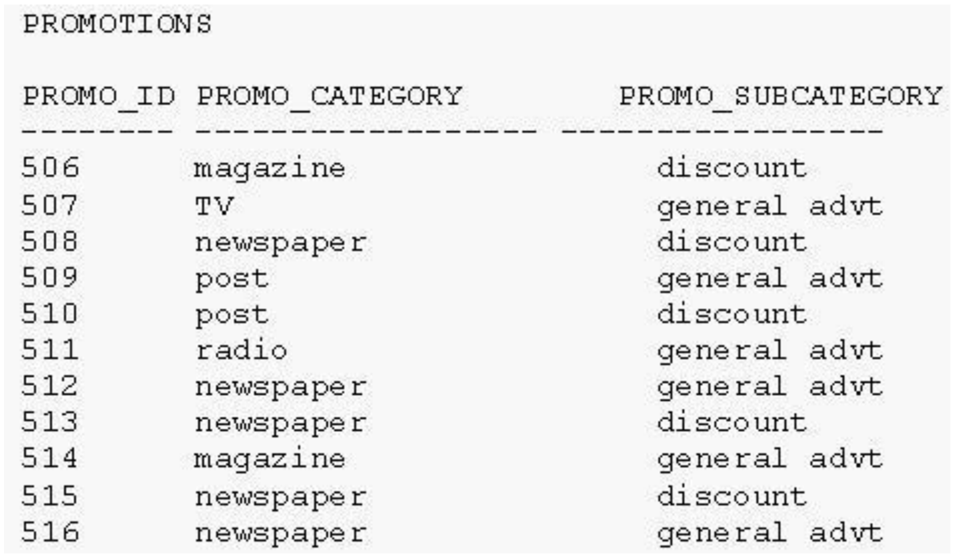
What would be the outcome of the above SQL statement?

A. It displays prod IDs in the promo with the lowest cost.
B. It displays prod IDs in the promos with the lowest cost in the same time interval.
C. It displays prod IDs in the promos with the highest cost in the same time interval.
D. It displays prod IDs in the promos with cost less than the highest cost in the same time interval.

Answer: D
Section: (none)

Explanation/Reference:

QUESTION 150
View the Exhibit and examine the data in the PROMOTIONS table.



```
PROMOTIONS

PROMO_ID PROMO_CATEGORY        PROMO_SUBCATEGORY
-------- ------------------ -----------------
506      magazine                 discount
507      TV                       general advt
508      newspaper                discount
509      post                     general advt
510      post                     discount
511      radio                    general advt
512      newspaper                general advt
513      newspaper                discount
514      magazine                 general advt
515      newspaper                discount
516      newspaper                general advt
```

You need to display all promo categories that do not have 'discount' in their subcategory.

Which two SQL statements give the required result? (Choose two.)

A. SELECT promo_category
FROM promotions
MINUS
SELECT promo_category
FROM promotions
WHERE promo_subcategory = 'discount';

B. SELECT promo_category
FROM promotions
INTERSECT
SELECT promo_category
FROM promotions
WHERE promo_subcategory = 'discount';

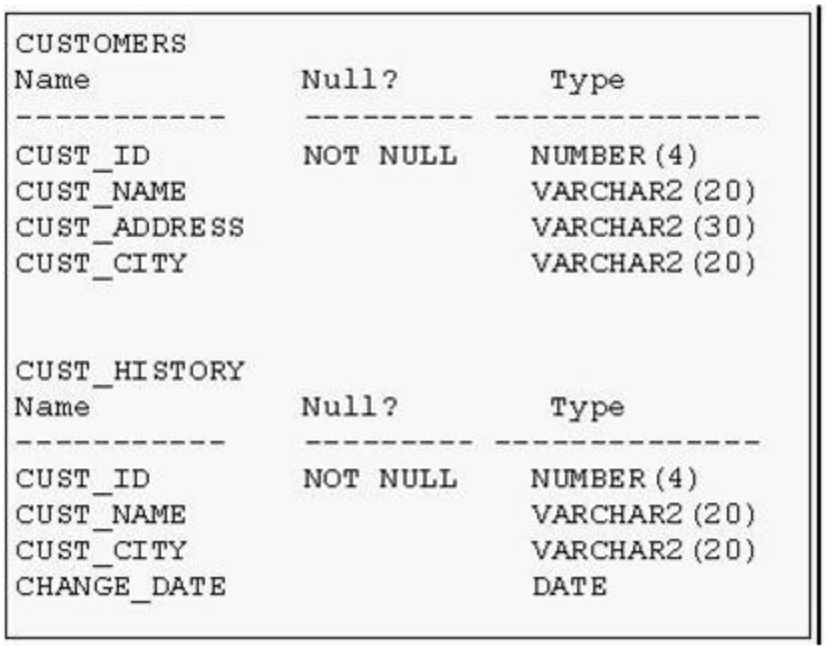
C. SELECT promo_category
FROM promotions
MINUS
SELECT promo_category
FROM promotions
WHERE promo_subcategory <> 'discount';

D. SELECT promo_category
FROM promotions
INTERSECT
SELECT promo_category
FROM promotions
WHERE promo_subcategory <> 'discount';

Answer: AD
Section: (none)

Explanation/Reference:

QUESTION 151
View the Exhibit and examine the structure of the CUSTOMERS and CUST_HISTORY tables.



```
CUSTOMERS
Name             Null?        Type
-----------      --------- -------------
CUST_ID          NOT NULL    NUMBER(4)
CUST_NAME                    VARCHAR2(20)
CUST_ADDRESS                 VARCHAR2(30)
CUST_CITY                    VARCHAR2(20)


CUST_HISTORY
Name             Null?        Type
-----------      --------- -------------
CUST_ID          NOT NULL    NUMBER(4)
CUST_NAME                    VARCHAR2(20)
CUST_CITY                    VARCHAR2(20)
CHANGE_DATE                  DATE
```

The CUSTOMERS table contains the current location of all currently active customers.
The CUST_HISTORY table stores historical details relating to any changes in the location of all current as well as previous customers who are no longer active with the company.

You need to find those customers who have never changed their address.

Which SET operator would you use to get the required output?

A. MINUS
B. UNION
C. INTERSECT
D. UNION ALL

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 152
Which statement is true regarding the UNION operator?

A. By default, the output is not sorted.
B. NULL values are not ignored during duplicate checking.
C. Names of all columns must be identical across all SELECT statements.
D. The number of columns selected in all SELECT statements need not be the same.

Answer: B
Section: (none)

Explanation/Reference:
The columns in the queries that make up a compound query can have different names, but the output result set will use the names of the columns in the first query.
A compound query will by default return **rows sorted** across all the columns, from left to right. The only exception is UNION ALL, where the rows will not be sorted. The only place where an ORDER BY clause is permitted is at the end of the compound query.

Oracle Server and Set Operators

• Duplicate rows are automatically eliminated except in UNION ALL.
• Column names from the first query appear in the result.
• The output is sorted in ascending order by default except in UNION ALL.

QUESTION 153
View the Exhibits and examine the structures of the PRODUCTS and SALES tables.

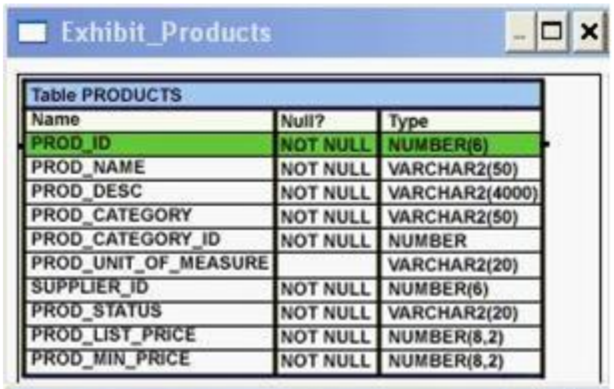

Exhibit_Products

Table PRODUCTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

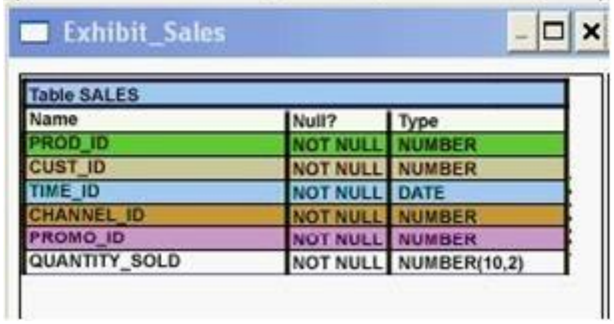

Exhibit_Sales

Table SALES

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
CUST_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
CHANNEL_ID	NOT NULL	NUMBER
PROMO_ID	NOT NULL	NUMBER
QUANTITY_SOLD	NOT NULL	NUMBER(10,2)

Which two SQL statements would give the same output? (Choose two.)

A. SELECT prod_id FROM products
   INTERSECT
   SELECT prod_id FROM sales;
B. SELECT prod_id FROM products
   MINUS
   SELECT prod_id FROM sales;
C. SELECT DISTINCT p.prod_id
   FROM products p JOIN sales s
   ON p.prod_id=s.prod_id;
D. SELECT DISTINCT p.prod_id
   FROM products p JOIN sales s
   ON p.prod_id <> s.prod_id;

Answer: AC
Section: (none)

Explanation/Reference:

QUESTION 154
View the Exhibit and evaluate structures of the SALES, PRODUCTS, and COSTS tables.

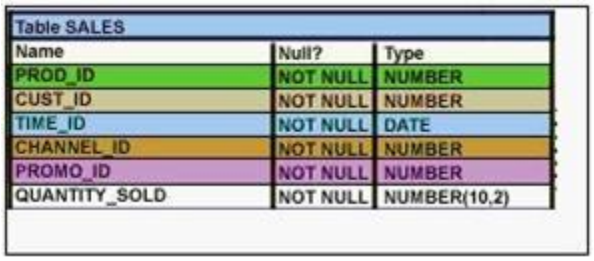


Table SALES

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
CUST_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
CHANNEL_ID	NOT NULL	NUMBER
PROMO_ID	NOT NULL	NUMBER
QUANTITY_SOLD	NOT NULL	NUMBER(10,2)

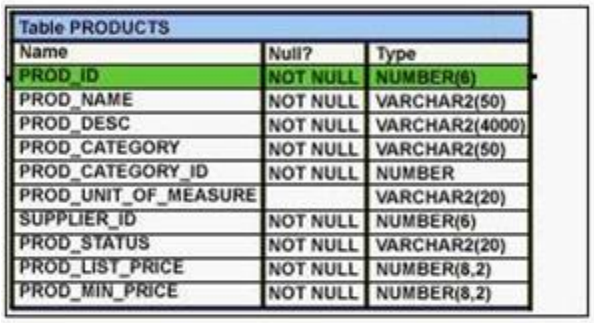


Table PRODUCTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

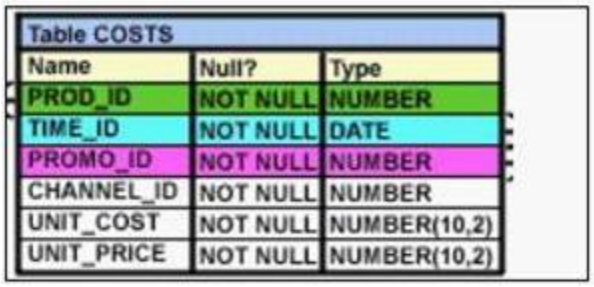


Table COSTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
PROMO_ID	NOT NULL	NUMBER
CHANNEL_ID	NOT NULL	NUMBER
UNIT_COST	NOT NULL	NUMBER(10,2)
UNIT_PRICE	NOT NULL	NUMBER(10,2)

Evaluate the following SQL statement:

```
SQL>SELECT prod_id FROM products
INTERSECT
SELECT prod_id FROM sales
MINUS
SELECT prod_id FROM costs;
```

Which statement is true regarding the above compound query?

A. It produces an error.
B. It shows products that were sold and have a cost recorded.
C. It shows products that were sold but have no cost recorded.
D. It shows products that have a cost recorded irrespective of sales.

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 155
Evaluate the following SQL statement:

```
SQL> SELECT promo_id, promo_category
FROM promotions
WHERE promo_category = 'Internet' ORDER BY 2 DESC
UNION
SELECT promo_id, promo_category
FROM promotions
```

```
WHERE promo_category = 'TV'
UNION
SELECT promo_id, promo_category
FROM promotions
WHERE promo_category ='Radio';
```

Which statement is true regarding the outcome of the above query?

A. It executes successfully and displays rows in the descending order of PROMO_CATEGORY.
B. It produces an error because positional notation cannot be used in the ORDER BY clause with SET operators.
C. It executes successfully but ignores the ORDER BY clause because it is not located at the end of the compound statement.
D. It produces an error because the ORDER BY clause should appear only at the end of a compound query-that is, with the last SELECT statement.

Answer: D
Section: (none)

Explanation/Reference:
Using the ORDER BY Clause in Set Operations
The ORDER BY clause can appear only once at the end of the compound query.
Component queries cannot have individual ORDER BY clauses.
The ORDER BY clause recognizes only the columns of the first SELECT query.
By default, the first column of the first SELECT query is used to sort the output in an ascending order.

QUESTION 156
Evaluate the following SQL statement:

```
SQL> SELECT cust_id, cust_last_name "Last Name"
FROM customers
WHERE country_id = 10
UNION
SELECT cust_id CUST_NO, cust_last_name
FROM customers
WHERE country_id = 30;
```

Which ORDER BY clauses are valid for the above query? (Choose all that apply.)

A. ORDER BY 2,1
B. ORDER BY CUST_NO
C. ORDER BY 2,cust_id
D. ORDER BY "CUST_NO"
E. ORDER BY "Last Name"

Answer: ACE
Section: (none)

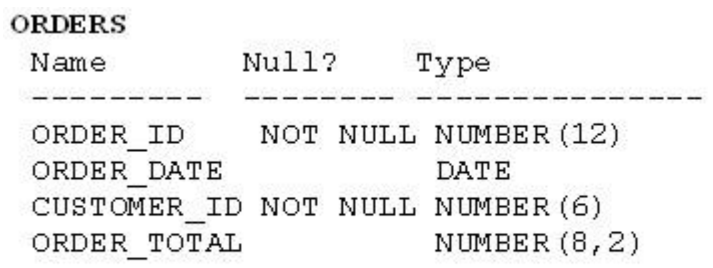
Explanation/Reference:
Using the ORDER BY Clause in Set Operations
- The ORDER BY clause can appear only once at the end of the compound query.
- Component queries cannot have individual ORDER BY clauses.
- The ORDER BY clause recognizes only the columns of the first SELECT query.
- By default, the first column of the first SELECT query is used to sort the output in an ascending order.

QUESTION 157
View the Exhibit and examine the structure of the ORDERS and CUSTOMERS tables.

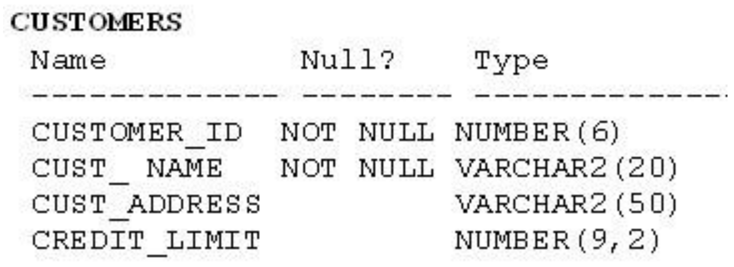
Evaluate the following SQL command:

```
SQL> SELECT o.order_id, c.cust_name, o.order_total, c.credit_limit
FROM orders o JOIN customers c
USING (customer_id)
WHERE o.order_total > c.credit_limit
FOR UPDATE
ORDER BY o.order_id;
```

Which two statements are true regarding the outcome of the above query? (Choose two.)



```
ORDERS
 Name         Null?     Type
 ---------   -------- ---------------
 ORDER_ID     NOT NULL NUMBER(12)
 ORDER_DATE            DATE
 CUSTOMER_ID NOT NULL NUMBER(6)
 ORDER_TOTAL           NUMBER(8,2)
```



```
CUSTOMERS
 Name            Null?      Type
 ------------- -------- -------------
 CUSTOMER_ID   NOT NULL NUMBER(6)
 CUST_ NAME    NOT NULL VARCHAR2(20)
 CUST_ADDRESS           VARCHAR2(50)
 CREDIT_LIMIT           NUMBER(9,2)
```

A. It locks all the rows that satisfy the condition in the statement.
B. It locks only the columns that satisfy the condition in both the tables.
C. The locks are released only when a COMMIT or ROLLBACK is issued.
D. The locks are released after a DML statement is executed on the locked rows.

Answer: AC
Section: (none)

Explanation/Reference:

FOR UPDATE Clause in a SELECT Statement
• Locks the rows in the EMPLOYEES table where job_id is SA_REP.
• Lock is released only when you issue a ROLLBACK or a COMMIT.
• If the SELECT statement attempts to lock a row that is locked by another user, the database waits until the row is available, and then returns the results of the SELECTstatement

```
SELECT employee_id, salary, commission_pct, job_id
FROM employees
WHERE job_id = 'SA_REP'
FOR UPDATE
ORDER BY employee_id;
```

QUESTION 158
Which statements are true regarding the FOR UPDATE clause in a SELECT statement? (Choose all that apply.)

A. It locks only the columns specified in the SELECT list.

B. It locks the rows that satisfy the condition in the SELECT statement.
C. It can be used only in SELECT statements that are based on a single table.
D. It can be used in SELECT statements that are based on a single or multiple tables.
E. After it is enforced by a SELECT statement, no other query can access the same rows until a COMMIT or ROLLBACK is issued.

Answer: BD
Section: (none)

Explanation/Reference:
FOR UPDATE Clause in a SELECT Statement
Locks the rows in the EMPLOYEES table where job_id is SA_REP.
Lock is released only when you issue a ROLLBACK or a COMMIT.
If the SELECT statement attempts to lock a row that is locked by another user, the database waits until the row is available, and then returns the results of the SELECT statement.
FOR UPDATE Clause in a SELECT Statement
When you issue a SELECT statement against the database to query some records, no locks are placed on the selected rows. In general, this is required because the number of records locked at any given time is (by default) kept to the absolute minimum: only those records that have been changed but not yet committed are locked. Even then, others will be able to read those records as they appeared before the change (the "before image" of the data). There are times, however, when you may want to lock a set of records even before you change them in your program.
Oracle offers the FOR UPDATE clause of the SELECT statement to perform this locking. When you issue a SELECT...FOR UPDATE statement, the relational database management system (RDBMS) automatically obtains exclusive row-level locks on all the rows identified by the SELECT statement, thereby holding the records "for your changes only." No one else will be able to change any of these records until you perform a ROLLBACK or a COMMIT.
You can append the optional keyword NOWAIT to the FOR UPDATE clause to tell the Oracle server not to wait if the table has been locked by another user. In this case, control will be returned immediately to your program or to your SQL Developer environment so that you can perform other work, or simply wait for a period of time before trying again. Without the NOWAIT clause, your process will block until the table is available, when the locks are released by the other user through the issue of a COMMIT or a ROLLBACK command.

QUESTION 159
View the Exhibit and examine the structure of the CUSTOMERS table.

NEW_CUSTOMERS is a new table with the columns CUST_ID, CUST_NAME and CUST_CITY that have the same data types and size as the corresponding columns in the CUSTOMERS table.

Evaluate the following INSERT statement:

```
INSERT INTO new_customers (cust_id, cust_name, cust_city) VALUES(SELECT
cust_id,cust_first_name||' '||cust_last_name,cust_city
FROM customers
WHERE cust_id > 23004);
```

The INSERT statement fails when executed. What could be the reason?

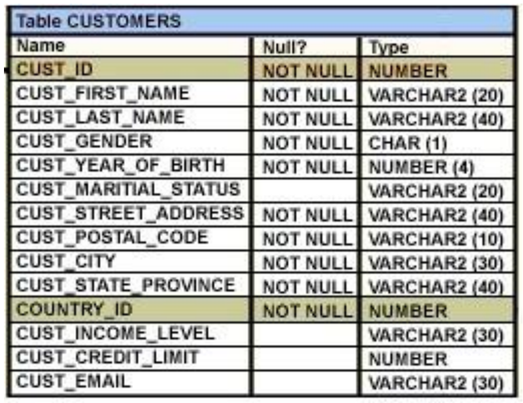


Table CUSTOMERS

Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

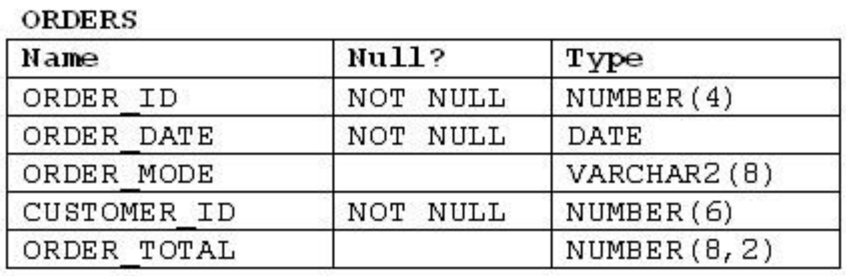
A. The VALUES clause cannot be used in an INSERT with a subquery.
B. Column names in the NEW_CUSTOMERS and CUSTOMERS tables do not match.
C. The WHERE clause cannot be used in a subquery embedded in an INSERT statement.
D. The total number of columns in the NEW_CUSTOMERS table does not match the total number of columns in the CUSTOMERS table.

Answer: A
Section: (none)

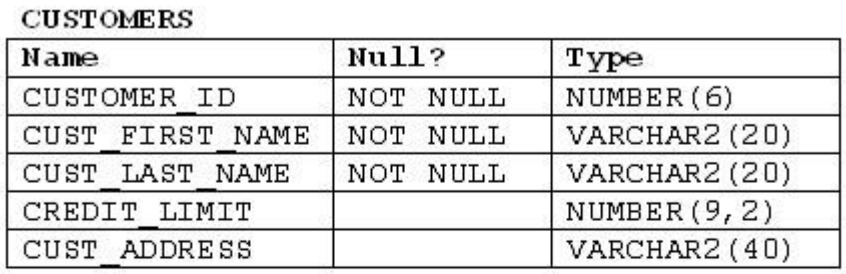
Explanation/Reference:
Copying Rows from Another Table
Write your INSERT statement with a subquery:
Do not use the VALUES clause.
Match the number of columns in the INSERT clause to those in the subquery.
Inserts all the rows returned by the subquery in the table, sales_reps.

QUESTION 160
View the Exhibit and examine the structure of ORDERS and CUSTOMERS tables.

There is only one customer with the cust_last_name column having value Roberts. Which INSERT statement should be used to add a row into the ORDERS table for the customer whose CUST_LAST_NAME is Roberts and CREDIT_LIMIT is 600?


ORDERS

Name	Null?	Type
ORDER_ID	NOT NULL	NUMBER(4)
ORDER_DATE	NOT NULL	DATE
ORDER_MODE		VARCHAR2(8)
CUSTOMER_ID	NOT NULL	NUMBER(6)
ORDER_TOTAL		NUMBER(8,2)


CUSTOMERS

Name	Null?	Type
CUSTOMER_ID	NOT NULL	NUMBER(6)
CUST_FIRST_NAME	NOT NULL	VARCHAR2(20)
CUST_LAST_NAME	NOT NULL	VARCHAR2(20)
CREDIT_LIMIT		NUMBER(9,2)
CUST_ADDRESS		VARCHAR2(40)

A. INSERT INTO orders
   VALUES (1,'10-mar-2007', 'direct',
   (SELECT customer_id
   FROM customers
   WHERE cust_last_name='Roberts' AND
   credit_limit=600), 1000);
B. INSERT INTO orders (order_id,order_date,order_mode,
   (SELECT customer_id
   FROM customers
   WHERE cust_last_name='Roberts' AND
   credit_limit=600),order_total)
   VALUES(1,'10-mar-2007', 'direct', &&customer_id, 1000);
C. INSERT INTO(SELECT o.order_id, o.order_date,o.order_mode,c.customer_id, o.order_total
   FROM orders o, customers c
   WHERE o.customer_id = c.customer_id
   AND c.cust_last_name='Roberts' ANDc.credit_limit=600)
   VALUES (1,'10-mar-2007', 'direct',(SELECT customer_id
   FROM customers
   WHERE cust_last_name='Roberts' AND
   credit_limit=600), 1000);
D. INSERT INTO orders (order_id,order_date,order_mode,
   (SELECT customer_id
   FROM customers
   WHERE cust_last_name='Roberts' AND
   credit_limit=600),order_total)
   VALUES(1,'10-mar-2007', 'direct', &customer_id, 1000);

Answer: A
Section: (none)

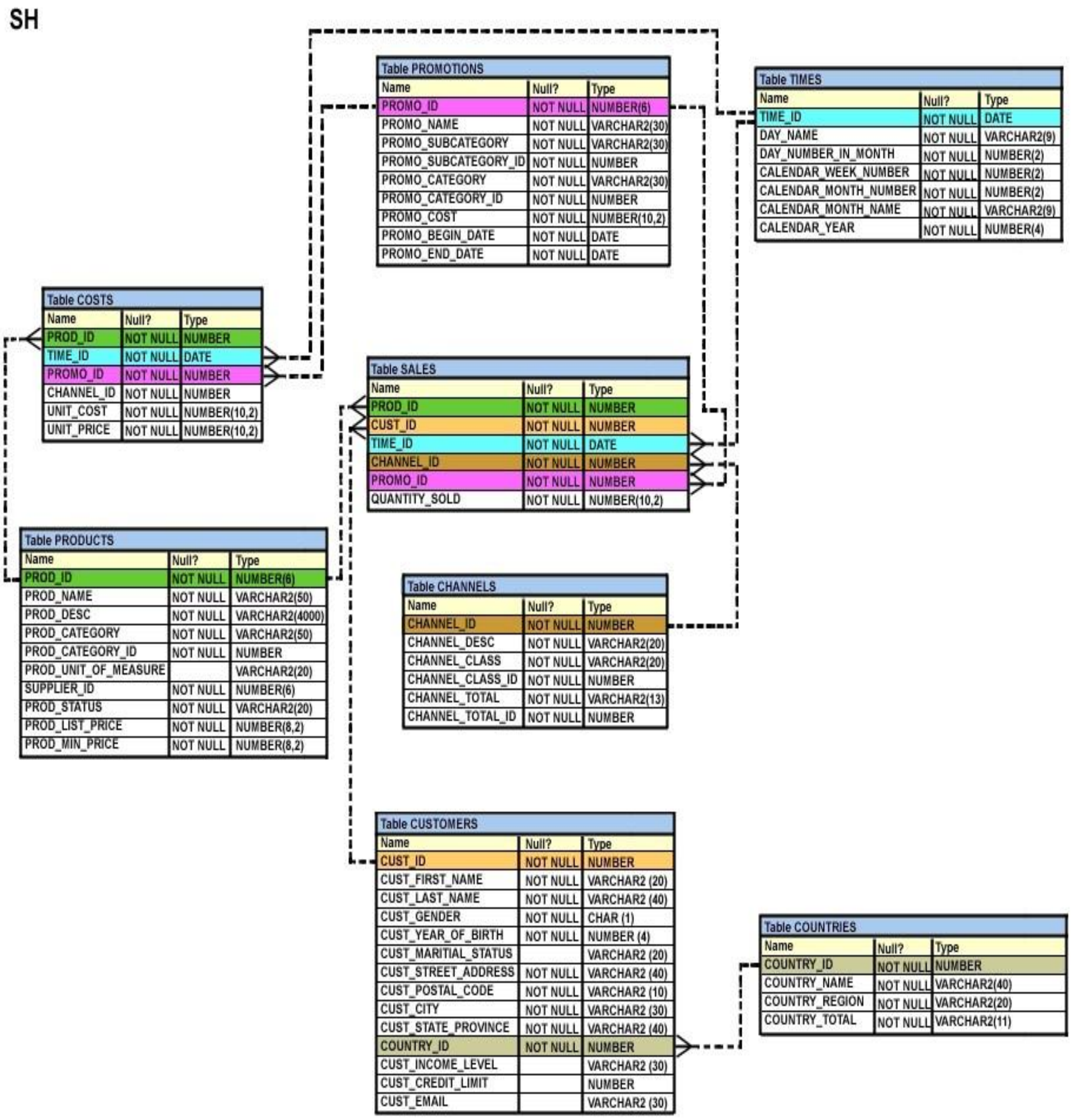
Explanation/Reference:

QUESTION 161
View the exhibit and examine the description for the SALES and CHANNELS tables.

You issued the following SQL statement to insert a row in the SALES table:

```
INSERT INTO sales VALUES
(23, 2300, SYSDATE, (SELECT channel_id
FROM channels
WHERE channel_desc='Direct Sales'), 12, 1, 500);
```

Which statement is true regarding the execution of the above statement?


SH
Table PROMOTIONS
Name Null? Type
PROMO_ID NOT NULL NUMBER(6)
PROMO_NAME NOT NULL VARCHAR2(30)
PROMO_SUBCATEGORY NOT NULL VARCHAR2(30)
PROMO_SUBCATEGORY_ID NOT NULL NUMBER
PROMO_CATEGORY NOT NULL VARCHAR2(30)
PROMO_CATEGORY_ID NOT NULL NUMBER
PROMO_COST NOT NULL NUMBER(10,2)
PROMO_BEGIN_DATE NOT NULL DATE
PROMO_END_DATE NOT NULL DATE
Table TIMES
Name Null? Type
TIME_ID NOT NULL DATE
DAY_NAME NOT NULL VARCHAR2(9)
DAY_NUMBER_IN_MONTH NOT NULL NUMBER(2)
CALENDAR_WEEK_NUMBER NOT NULL NUMBER(2)
CALENDAR_MONTH_NUMBER NOT NULL NUMBER(2)
CALENDAR_MONTH_NAME NOT NULL VARCHAR2(9)
CALENDAR_YEAR NOT NULL NUMBER(4)
Table COSTS
Name Null? Type
PROD_ID NOT NULL NUMBER
TIME_ID NOT NULL DATE
PROMO_ID NOT NULL NUMBER
CHANNEL_ID NOT NULL NUMBER
UNIT_COST NOT NULL NUMBER(10,2)
UNIT_PRICE NOT NULL NUMBER(10,2)
Table SALES
Name Null? Type
PROD_ID NOT NULL NUMBER
CUST_ID NOT NULL NUMBER
TIME_ID NOT NULL DATE
CHANNEL_ID NOT NULL NUMBER
PROMO_ID NOT NULL NUMBER
QUANTITY_SOLD NOT NULL NUMBER(10,2)
Table PRODUCTS
Name Null? Type
PROD_ID NOT NULL NUMBER(6)
PROD_NAME NOT NULL VARCHAR2(50)
PROD_DESC NOT NULL VARCHAR2(4000)
PROD_CATEGORY NOT NULL VARCHAR2(50)
PROD_CATEGORY_ID NOT NULL NUMBER
PROD_UNIT_OF_MEASURE VARCHAR2(20)
SUPPLIER_ID NOT NULL NUMBER(6)
PROD_STATUS NOT NULL VARCHAR2(20)
PROD_LIST_PRICE NOT NULL NUMBER(8,2)
PROD_MIN_PRICE NOT NULL NUMBER(8,2)
Table CHANNELS
Name Null? Type
CHANNEL_ID NOT NULL NUMBER
CHANNEL_DESC NOT NULL VARCHAR2(20)
CHANNEL_CLASS NOT NULL VARCHAR2(20)
CHANNEL_CLASS_ID NOT NULL NUMBER
CHANNEL_TOTAL NOT NULL VARCHAR2(13)
CHANNEL_TOTAL_ID NOT NULL NUMBER
Table CUSTOMERS
Name Null? Type
CUST_ID NOT NULL NUMBER
CUST_FIRST_NAME NOT NULL VARCHAR2 (20)
CUST_LAST_NAME NOT NULL VARCHAR2 (40)
CUST_GENDER NOT NULL CHAR (1)
CUST_YEAR_OF_BIRTH NOT NULL NUMBER (4)
CUST_MARITIAL_STATUS VARCHAR2 (20)
CUST_STREET_ADDRESS NOT NULL VARCHAR2 (40)
CUST_POSTAL_CODE NOT NULL VARCHAR2 (10)
CUST_CITY NOT NULL VARCHAR2 (30)
CUST_STATE_PROVINCE NOT NULL VARCHAR2 (40)
COUNTRY_ID NOT NULL NUMBER
CUST_INCOME_LEVEL VARCHAR2 (30)
CUST_CREDIT_LIMIT NUMBER
CUST_EMAIL VARCHAR2 (30)
Table COUNTRIES
Name Null? Type
COUNTRY_ID NOT NULL NUMBER
COUNTRY_NAME NOT NULL VARCHAR2(40)
COUNTRY_REGION NOT NULL VARCHAR2(20)
COUNTRY_TOTAL NOT NULL VARCHAR2(11)

A. The statement will execute and the new row will be inserted in the SALES table.
B. The statement will fail because subquery cannot be used in the VALUES clause.
C. The statement will fail because the VALUES clause is not required with subquery.
D. The statement will fail because subquery in the VALUES clause is not enclosed with in single quotation marks .

Answer: A
Section: (none)

Explanation/Reference:

QUESTION 162
View the Exhibit and examine the structure of the PRODUCTS, SALES, and SALE_SUMMARY tables.

SALE_VW is a view created using the following command :

```
SQL>CREATE VIEW sale_vw AS
SELECT prod_id, SUM(quantity_sold) QTY_SOLD
FROM sales GROUP BY prod_id;
```

You issue the following command to add a row to the SALE_SUMMARY table :

```
SQL>INSERT INTO sale_summary
SELECT prod_id, prod_name, qty_sold FROM sale_vw JOIN products
USING (prod_id) WHERE prod_id = 16;
```

What is the outcome?

A. It executes successfully.
B. It gives an error because a complex view cannot be used to add data into the SALE_SUMMARY table.
C. It gives an error because the column names in the subquery and the SALE_SUMMARY table do not match.
D. It gives an error because the number of columns to be inserted does not match with the number of columns in the SALE_SUMMARY table.

Answer: D
Section: (none)

Explanation/Reference:

QUESTION 163
View the Exhibit and examine the description for the CUSTOMERS table.

You want to update the CUST_CREDIT_LIMIT column to NULL for all the customers, where CUST_INCOME_LEVEL has NULL in the CUSTOMERS table. Which SQL statement will accomplish the task?

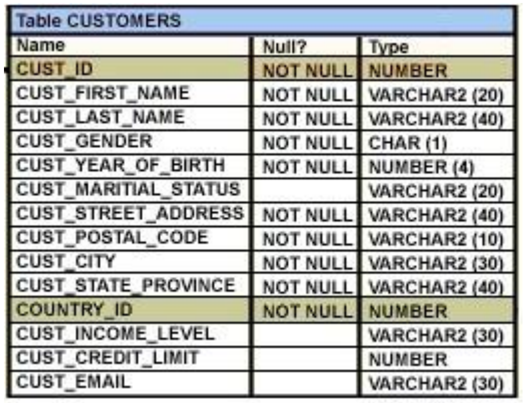


Table CUSTOMERS		
Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

A. UPDATE customers
SET cust_credit_limit = NULL
WHERE CUST_INCOME_LEVEL = NULL;
B. UPDATE customers
SET cust_credit_limit = NULL
WHERE cust_income_level IS NULL;
C. UPDATE customers
SET cust_credit_limit = TO_NUMBER(NULL)
WHERE cust_income_level = TO_NUMBER(NULL);
D. UPDATE customers

```
SET cust_credit_limit = TO_NUMBER(' ',9999)
WHERE cust_income_level IS NULL;
```

Answer: B
Section: (none)

Explanation/Reference:

QUESTION 164
View the Exhibit and examine the structure of CUSTOMERS and SALES tables.

Evaluate the following SQL statement:

```
UPDATE (SELECT prod_id, cust_id, quantity_sold, time_id
FROM sales)
SET time_id = '22-MAR-2007'
WHERE cust_id = (SELECT cust_id
FROM customers
WHERE cust_last_name = 'Roberts' AND
credit_limit = 600);
```

Which statement is true regarding the execution of the above UPDATE statement?

SH

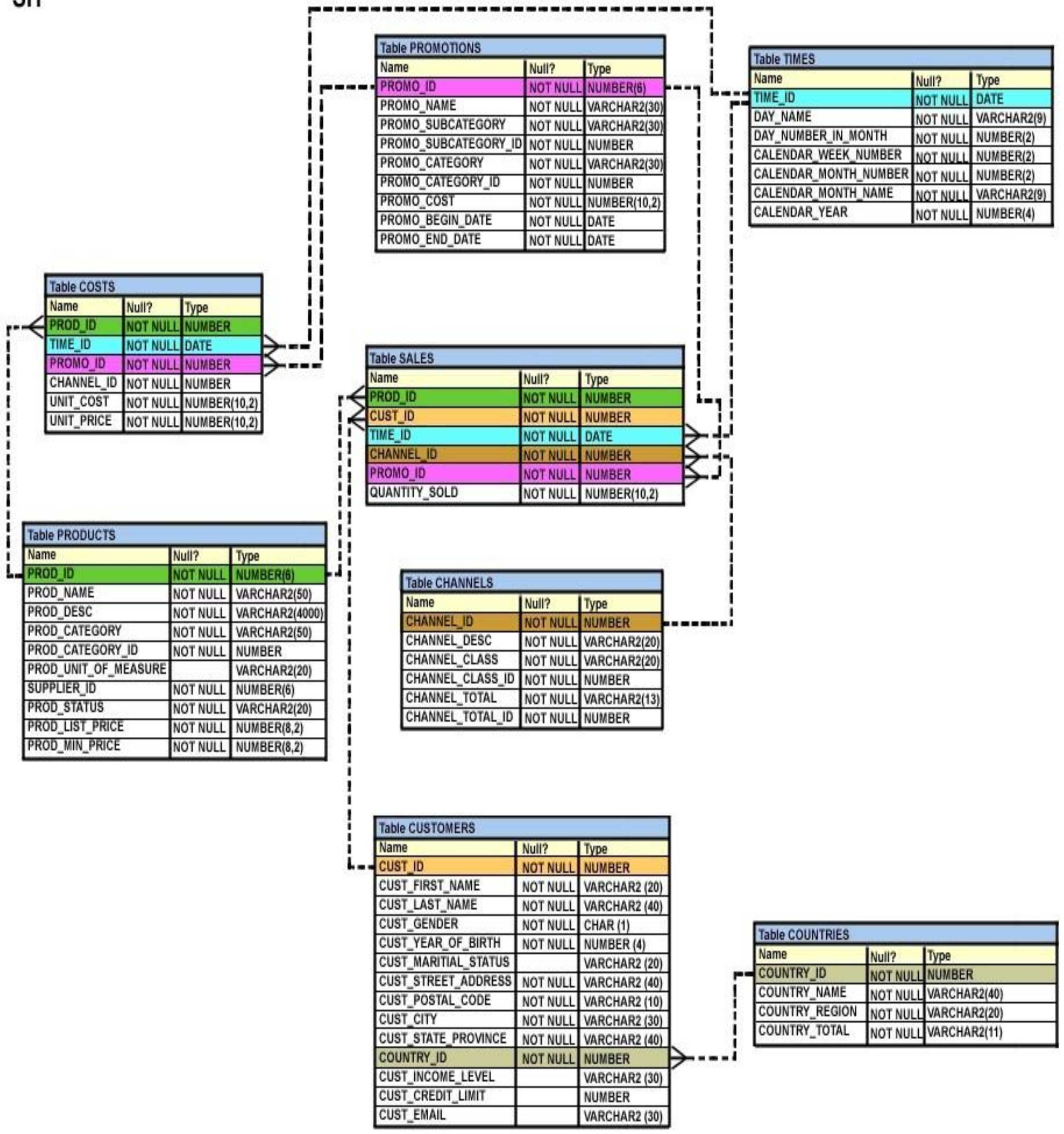


Table PROMOTIONS

Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

Table TIMES

Name	Null?	Type
TIME_ID	NOT NULL	DATE
DAY_NAME	NOT NULL	VARCHAR2(9)
DAY_NUMBER_IN_MONTH	NOT NULL	NUMBER(2)
CALENDAR_WEEK_NUMBER	NOT NULL	NUMBER(2)
CALENDAR_MONTH_NUMBER	NOT NULL	NUMBER(2)
CALENDAR_MONTH_NAME	NOT NULL	VARCHAR2(9)
CALENDAR_YEAR	NOT NULL	NUMBER(4)

Table COSTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
PROMO_ID	NOT NULL	NUMBER
CHANNEL_ID	NOT NULL	NUMBER
UNIT_COST	NOT NULL	NUMBER(10,2)
UNIT_PRICE	NOT NULL	NUMBER(10,2)

Table SALES

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
CUST_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
CHANNEL_ID	NOT NULL	NUMBER
PROMO_ID	NOT NULL	NUMBER
QUANTITY_SOLD	NOT NULL	NUMBER(10,2)

Table PRODUCTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

Table CHANNELS

Name	Null?	Type
CHANNEL_ID	NOT NULL	NUMBER
CHANNEL_DESC	NOT NULL	VARCHAR2(20)
CHANNEL_CLASS	NOT NULL	VARCHAR2(20)
CHANNEL_CLASS_ID	NOT NULL	NUMBER
CHANNEL_TOTAL	NOT NULL	VARCHAR2(13)
CHANNEL_TOTAL_ID	NOT NULL	NUMBER

Table CUSTOMERS

Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

Table COUNTRIES

Name	Null?	Type
COUNTRY_ID	NOT NULL	NUMBER
COUNTRY_NAME	NOT NULL	VARCHAR2(40)
COUNTRY_REGION	NOT NULL	VARCHAR2(20)
COUNTRY_TOTAL	NOT NULL	VARCHAR2(11)

A. It would not execute because two tables cannot be used in a single UPDATE statement.
B. It would not execute because the SELECT statement cannot be used in place of the table name.
C. It would execute and restrict modifications to only the columns specified in the SELECT statement.
D. It would not execute because a subquery cannot be used in the WHERE clause of an UPDATE statement.

Answer: C
Section: (none)

Explanation/Reference:
One UPDATE statement can change rows in only one table, but it can change any number of rows in that table.

QUESTION 165
View the Exhibit and examine the description for the CUSTOMERS table.

You want to update the CUST_INCOME_LEVEL and CUST_CREDIT_LIMIT columns for the customer with the CUST_ID 2360.
You want the value for the CUST_INCOME_LEVEL to have the same value as that of the customer with the CUST_ID 2560 and the CUST_CREDIT_LIMIT to have the same value as that of the
customer with CUST_ID 2566.

Which UPDATE statement will accomplish the task?

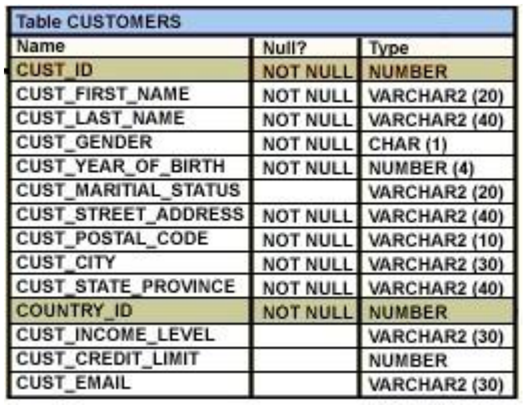


Table CUSTOMERS		
Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

A. UPDATE customers
SET cust_income_level = (SELECT cust_income_level
FROM customers
WHERE cust_id = 2560),
cust_credit_limit = (SELECT cust_credit_limit
FROM customers
WHERE cust_id = 2566)
WHERE cust_id=2360;

B. UPDATE customers
SET (cust_income_level,cust_credit_limit) = (SELECT
cust_income_level, cust_credit_limit
FROM customers
WHERE cust_id=2560 OR cust_id=2566)
WHERE cust_id=2360;

C. UPDATE customers
SET (cust_income_level,cust_credit_limit) = (SELECT
cust_income_level, cust_credit_limit
FROM customers
WHERE cust_id IN(2560, 2566)
WHERE cust_id=2360;

D. UPDATE customers
SET (cust_income_level,cust_credit_limit) = (SELECT
cust_income_level, cust_credit_limit
FROM customers
WHERE cust_id=2560 AND cust_id=2566)
WHERE cust_id=2360;

Answer: A
Section: (none)

Explanation/Reference:

Updating Two Columns with a Subquery

You can update multiple columns in the SET clause of an UPDATE statement by writing multiple subqueries. The syntax is as follows:

UPDATE **table**

```
SET column =
(SELECT column
FROM table
WHERE condition)
[ ,
column =
(SELECT column
FROM table
WHERE condition)]
[WHERE condition ] ;
```

QUESTION 166
View the Exhibit and examine the structures of the EMPLOYEES and DEPARTMENTS tables.

You want to update the EMPLOYEES table as follows:4 ? 4;
-Update only those employees who work in Boston or Seattle (locations 2900 and 2700).
-Set department_id for these employees to the department_id corresponding to London (location_id 2100).
-Set the employees' salary in location_id 2100 to 1.1 times the average salary of their department.
-Set the employees' commission in location_id 2100 to 1.5 times the average commission of their department.

You issue the following command:

```
SQL>UPDATE employees
SET department_id =
(SELECT department_id
FROM departments
WHERE location_id = 2100),
(salary, commission) =
(SELECT 1.1*AVG(salary), 1.5*AVG(commission)
FROM employees, departments
WHERE departments.location_id IN(2900,2700,2100))
WHERE department_id IN
(SELECT department_id
FROM departments
WHERE location_id = 2900
OR location_id = 2700)
```

What is the outcome?

A. It executes successfully and gives the correct result.
B. It executes successfully but does not give the correct result.
C. It generates an error because a subquery cannot have a join condition in an UPDATE statement.
D. It generates an error because multiple columns (SALARY, COMMISION) cannot be specified together in an UPDATE statement.

Answer: B
Section: (none)

Explanation/Reference:

QUESTION 167
Evaluate the following DELETE statement:

```
DELETE FROM sales;
```

There are no other uncommitted transactions on the SALES table.

Which statement is true about the DELETE statement?

A. It would not remove the rows if the table has a primary key.
B. It removes all the rows as well as the structure of the table.
C. It removes all the rows in the table and deleted rows can be rolled back.
D. It removes all the rows in the table and deleted rows cannot be rolled back.

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 168
View the Exhibit and examine the description of SALES and PROMOTIONS tables.

You want to delete rows from the SALES table, where the PROMO_NAME column in the PROMOTIONS table has either blowout sale or everyday low price as values.

Which DELETE statements are valid? (Choose all that apply.)

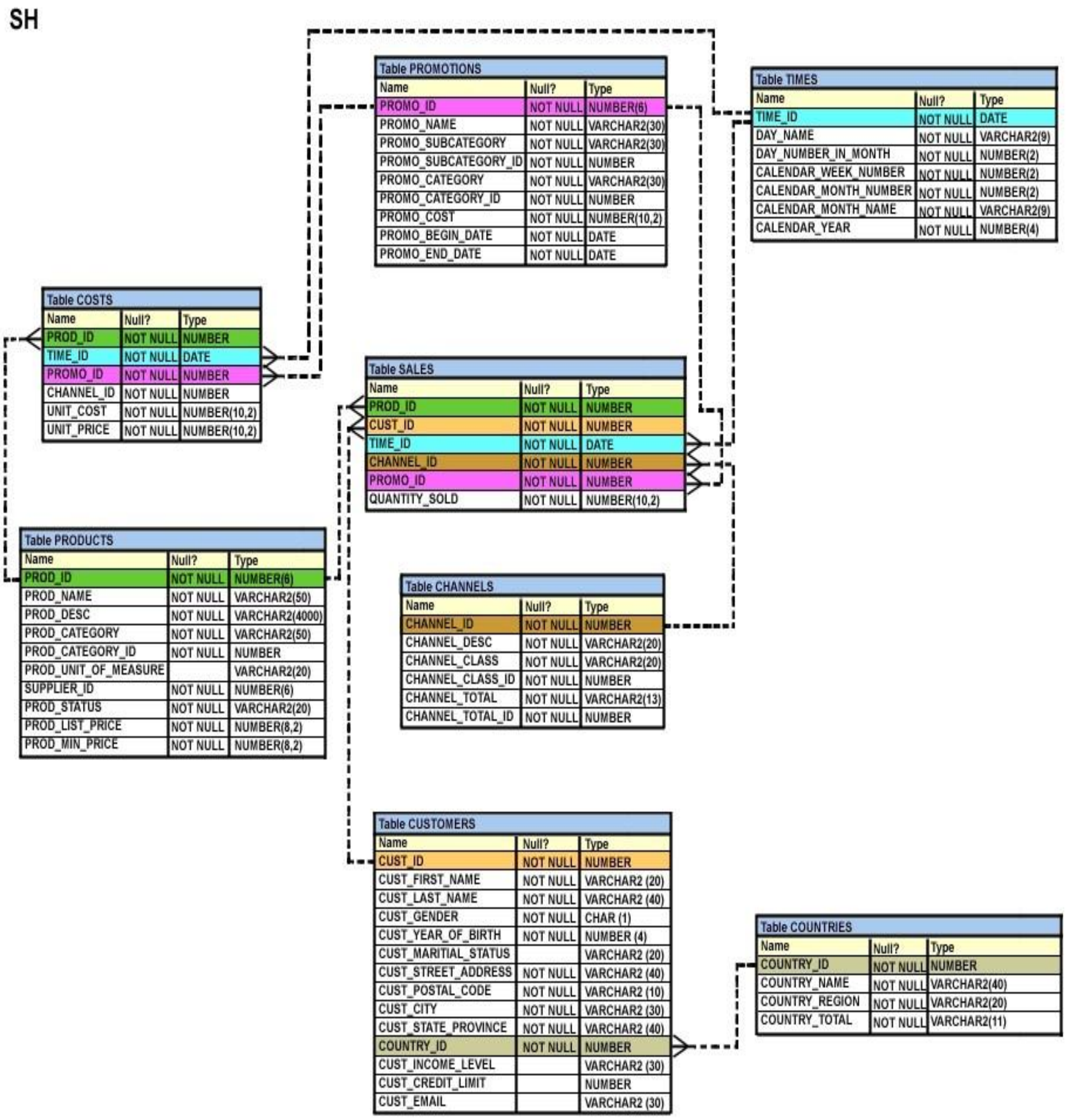


Table PROMOTIONS

Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

Table TIMES

Name	Null?	Type
TIME_ID	NOT NULL	DATE
DAY_NAME	NOT NULL	VARCHAR2(9)
DAY_NUMBER_IN_MONTH	NOT NULL	NUMBER(2)
CALENDAR_WEEK_NUMBER	NOT NULL	NUMBER(2)
CALENDAR_MONTH_NUMBER	NOT NULL	NUMBER(2)
CALENDAR_MONTH_NAME	NOT NULL	VARCHAR2(9)
CALENDAR_YEAR	NOT NULL	NUMBER(4)

Table COSTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
PROMO_ID	NOT NULL	NUMBER
CHANNEL_ID	NOT NULL	NUMBER
UNIT_COST	NOT NULL	NUMBER(10,2)
UNIT_PRICE	NOT NULL	NUMBER(10,2)

Table SALES

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
CUST_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
CHANNEL_ID	NOT NULL	NUMBER
PROMO_ID	NOT NULL	NUMBER
QUANTITY_SOLD	NOT NULL	NUMBER(10,2)

Table PRODUCTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

Table CHANNELS

Name	Null?	Type
CHANNEL_ID	NOT NULL	NUMBER
CHANNEL_DESC	NOT NULL	VARCHAR2(20)
CHANNEL_CLASS	NOT NULL	VARCHAR2(20)
CHANNEL_CLASS_ID	NOT NULL	NUMBER
CHANNEL_TOTAL	NOT NULL	VARCHAR2(13)
CHANNEL_TOTAL_ID	NOT NULL	NUMBER

Table CUSTOMERS

Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

Table COUNTRIES

Name	Null?	Type
COUNTRY_ID	NOT NULL	NUMBER
COUNTRY_NAME	NOT NULL	VARCHAR2(40)
COUNTRY_REGION	NOT NULL	VARCHAR2(20)
COUNTRY_TOTAL	NOT NULL	VARCHAR2(11)

A. DELETE
FROM sales
WHERE promo_id = (SELECT promo_id
FROM promotions
WHERE promo_name = 'blowout sale')
AND promo_id = (SELECT promo_id
FROM promotions
WHERE promo_name = 'everyday low price');

B. DELETE
FROM sales
WHERE promo_id = (SELECT promo_id
FROM promotions

WHERE promo_name = 'blowout sale')
OR promo_id = (SELECT promo_id
FROM promotions
WHERE promo_name = 'everyday low price');

C. DELETE
FROM sales
WHERE promo_id IN (SELECT promo_id
FROM promotions
WHERE promo_name = 'blowout sale'
OR promo_name = 'everyday low price');

D. D DELETE
FROM sales
WHERE promo_id IN (SELECT promo_id
FROM promotions
WHERE promo_name IN ('blowout sale','everyday low price'));

Answer: BCD
Section: (none)

Explanation/Reference:

QUESTION 169
View the Exhibit and examine the description for the PRODUCTS and SALES table.

PROD_ID is a primary key in the PRODUCTS table and foreign key in the SALES table.
You want to remove all the rows from the PRODUCTS table for which no sale was done for the last three years.

Which is the valid DELETE statement?

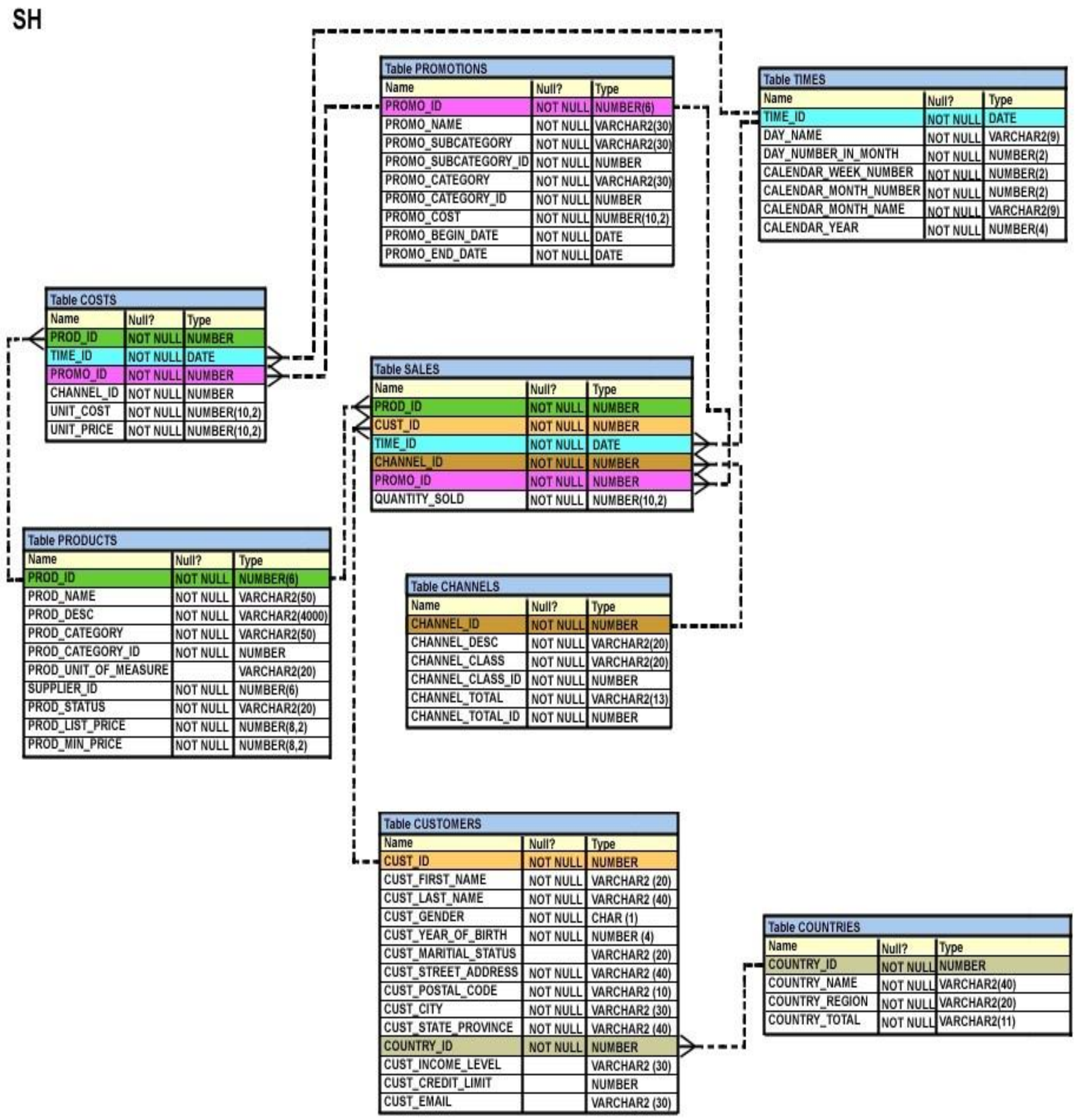

SH

Table PROMOTIONS

Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

Table TIMES

Name	Null?	Type
TIME_ID	NOT NULL	DATE
DAY_NAME	NOT NULL	VARCHAR2(9)
DAY_NUMBER_IN_MONTH	NOT NULL	NUMBER(2)
CALENDAR_WEEK_NUMBER	NOT NULL	NUMBER(2)
CALENDAR_MONTH_NUMBER	NOT NULL	NUMBER(2)
CALENDAR_MONTH_NAME	NOT NULL	VARCHAR2(9)
CALENDAR_YEAR	NOT NULL	NUMBER(4)

Table COSTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
PROMO_ID	NOT NULL	NUMBER
CHANNEL_ID	NOT NULL	NUMBER
UNIT_COST	NOT NULL	NUMBER(10,2)
UNIT_PRICE	NOT NULL	NUMBER(10,2)

Table SALES

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
CUST_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
CHANNEL_ID	NOT NULL	NUMBER
PROMO_ID	NOT NULL	NUMBER
QUANTITY_SOLD	NOT NULL	NUMBER(10,2)

Table PRODUCTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

Table CHANNELS

Name	Null?	Type
CHANNEL_ID	NOT NULL	NUMBER
CHANNEL_DESC	NOT NULL	VARCHAR2(20)
CHANNEL_CLASS	NOT NULL	VARCHAR2(20)
CHANNEL_CLASS_ID	NOT NULL	NUMBER
CHANNEL_TOTAL	NOT NULL	VARCHAR2(13)
CHANNEL_TOTAL_ID	NOT NULL	NUMBER

Table CUSTOMERS

Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

Table COUNTRIES

Name	Null?	Type
COUNTRY_ID	NOT NULL	NUMBER
COUNTRY_NAME	NOT NULL	VARCHAR2(40)
COUNTRY_REGION	NOT NULL	VARCHAR2(20)
COUNTRY_TOTAL	NOT NULL	VARCHAR2(11)

A. DELETE
FROM products
WHERE prod_id = (SELECT prod_id
FROM sales
WHERE time_id - 3*365 = SYSDATE);

B. DELETE
FROM products
WHERE prod_id = (SELECT prod_id
FROM sales
WHERE SYSDATE >= time_id - 3*365);

C. DELETE
FROM products

```
WHERE prod_id IN (SELECT prod_id
FROM sales
WHERE SYSDATE - 3*365 >= time_id);
```

D. DELETE
```
FROM products
WHERE prod_id IN (SELECT prod_id
FROM sales
WHERE time_id >= SYSDATE - 3*365 );
```

Answer: C
Section: (none)

Explanation/Reference:

QUESTION 170
Which two statements are true regarding the DELETE and TRUNCATE commands? (Choose two.)

A. DELETE can be used to remove only rows from only one table at a time.
B. DELETE can be used to remove only rows from multiple tables at a time.
C. DELETE can be used only on a table that is a parent of a referential integrity constraint.
D. DELETE can be used to remove data from specific columns as well as complete rows.
E. DELETE and TRUNCATE can be used on a table that is a parent of a referential integrity constraint having ON DELETE rule .

Answer: AE
Section: (none)

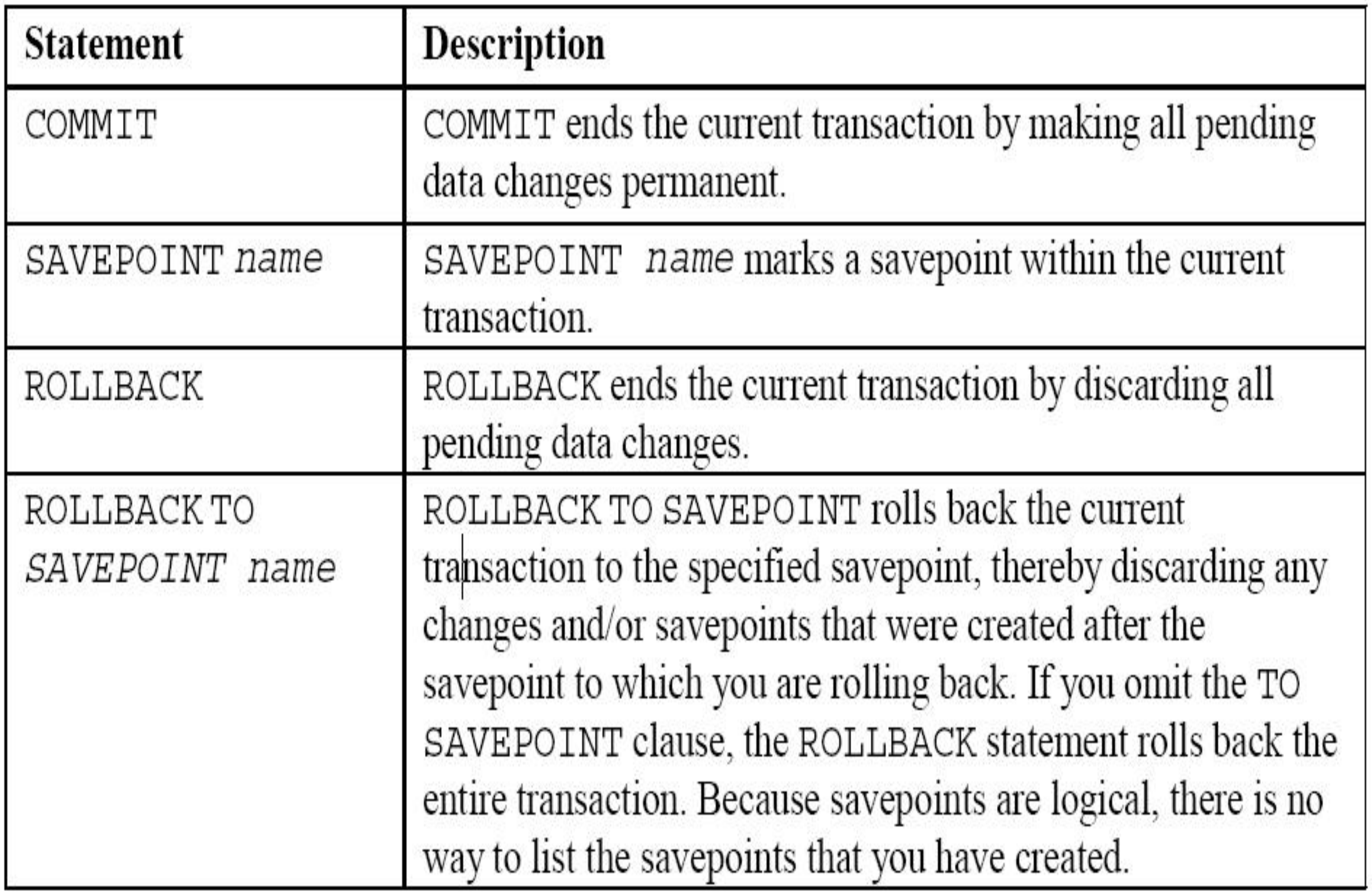
Explanation/Reference:
Transactions, consisting of INSERT, UPDATE, and DELETE (or even MERGE) commands can be made permanent (with a COMMIT) or reversed (with a ROLLBACK). A TRUNCATE
command, like any other DDL command, is immediately permanent: it can never be reversed.

The Transaction Control Statements

A transaction begins implicitly with the first DML statement. There is no command to explicitly start a transaction. The transaction continues through all subsequent DML statements issued by the session. These statements can be against any number of tables: a transaction is not restricted to one table. It terminates (barring any of the events listed in the previous section) when the session issues a COMMIT or ROLLBACK command. The SAVEPOINT command can be used to set markers that will stage the action of a ROLLBACK, but the same transaction remains in progress irrespective of the use of SAVEPOINT

Explicit Transaction Control Statements

You can control the logic of transactions by using the COMMIT, SAVEPOINT, and ROLLBACK statements.
Note: You cannot COMMIT to a SAVEPOINT. SAVEPOINT is not ANSI-standard SQL.



Statement	Description
COMMIT	COMMIT ends the current transaction by making all pending data changes permanent.
SAVEPOINT *name*	SAVEPOINT *name* marks a savepoint within the current transaction.
ROLLBACK	ROLLBACK ends the current transaction by discarding all pending data changes.
ROLLBACK TO *SAVEPOINT name*	ROLLBACK TO SAVEPOINT rolls back the current transaction to the specified savepoint, thereby discarding any changes and/or savepoints that were created after the savepoint to which you are rolling back. If you omit the TO SAVEPOINT clause, the ROLLBACK statement rolls back the entire transaction. Because savepoints are logical, there is no way to list the savepoints that you have created.

QUESTION 171
Which three statements/commands would cause a transaction to end? (Choose three.)

A. COMMIT
B. SELECT
C. CREATE
D. ROLLBACK
E. SAVEPOINT

Answer: ACD
Section: (none)

Explanation/Reference:

QUESTION 172
The SQL statements executed in a user session are as follows:

```
SQL> CREATE TABLE product
(pcode NUMBER(2),
pname VARCHAR2(10));
SQL> INSERT INTO product VALUES (1, 'pen');
SQL> INSERT INTO product VALUES (2,'pencil');
SQL> SAVEPOINT a;
SQL> UPDATE product SET pcode = 10 WHERE pcode = 1;
SQL> SAVEPOINT b;
SQL> DELETE FROM product WHERE pcode = 2;
SQL> COMMIT;
SQL> DELETE FROM product WHERE pcode=10;
```

Which two statements describe the consequences of issuing the ROLLBACK TO SAVE POINT a command in the session? (Choose two.)

A. The rollback generates an error.
B. No SQL statements are rolled back.
C. Only the DELETE statements are rolled back.
D. Only the second DELETE statement is rolled back.
E. Both the DELETE statements and the UPDATE statement are rolled back.

Answer: AB
Section: (none)

Explanation/Reference:

QUESTION 173
When does a transaction complete? (Choose all that apply.)

A. when a DELETE statement is executed
B. when a ROLLBACK command is executed
C. when a PL/SQL anonymous block is executed
D. when a data definition language (DDL) statement is executed
E. when a TRUNCATE statement is executed after the pending transaction

Answer: BDE
Section: (none)

Explanation/Reference:

QUESTION 174
Which statement is true regarding transactions? (Choose all that apply.)

A. A transaction can consist only of a set of DML and DDL statements.
B. A part or an entire transaction can be undone by using ROLLBACK command .
C. A transaction consists of a set of DML or DCL statements.
D. A part or an entire transaction can be made permanent with a COMMIT.
E. A transaction can consist of only a set of queries or DML or DDL statements.

Answer: BC
Section: (none)

Explanation/Reference:

QUESTION 175
Which two statements are true regarding savepoints? (Choose two.)

A. Savepoints are effective only for COMMIT.
B. Savepoints may be used to ROLLBACK.
C. Savepoints can be used for only DML statements.
D. Savepoints are effective for both COMMIT and ROLLBACK.
E. Savepoints can be used for both DML and DDL statements.

Answer: BC
Section: (none)

Explanation/Reference:

QUESTION 176
View the Exhibit; e xamine the structure of the PROMOTIONS table.

Each promotion has a duration of at least seven days .

Your manager has asked you to generate a report, which provides the weekly cost for each promotion done to l date.

Which query would achieve the required result?

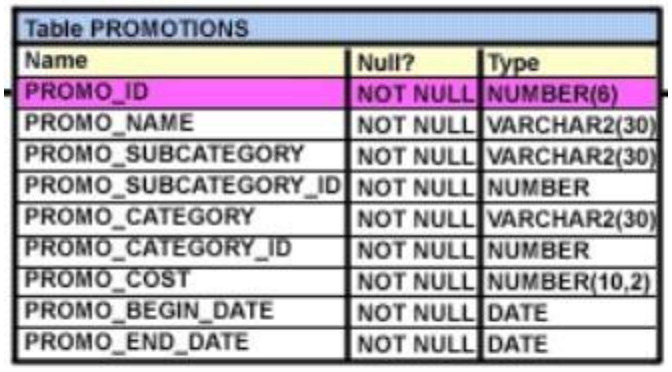


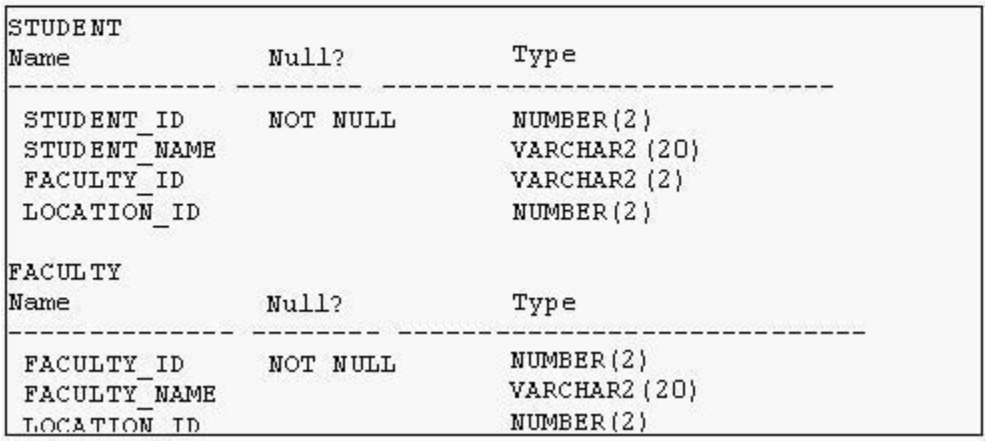
Table PROMOTIONS		
Name	Null?	Type
PROMO_ID	NOT NULL	NUMBER(6)
PROMO_NAME	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY	NOT NULL	VARCHAR2(30)
PROMO_SUBCATEGORY_ID	NOT NULL	NUMBER
PROMO_CATEGORY	NOT NULL	VARCHAR2(30)
PROMO_CATEGORY_ID	NOT NULL	NUMBER
PROMO_COST	NOT NULL	NUMBER(10,2)
PROMO_BEGIN_DATE	NOT NULL	DATE
PROMO_END_DATE	NOT NULL	DATE

A. SELECT promo_name, promo_cost/promo_end_date-promo_begin_date/7 FROM promotions;
B. SELECT promo_name,(promo_cost/promo_end_date-promo_begin_date)/7 FROM promotions;
C. SELECT promo_name, promo_cost/(promo_end_date-promo_begin_date/7) FROM promotions;
D. SELECT promo_name, promo_cost/((promo_end_date-promo_begin_date)/7) FROM promotions;

Answer: D
Section: (none)

Explanation/Reference:

QUESTION 177
View the Exhibit for the structure of the STUDENT and FACULTY tables.
You need to display the faculty name followed by the number of students handled by the faculty at the base location.



```
STUDENT
Name              Null?          Type
------------- -------- ----------------------------
 STUDENT_ID       NOT NULL       NUMBER(2)
 STUDENT_NAME                    VARCHAR2(20)
 FACULTY_ID                      VARCHAR2(2)
 LOCATION_ID                     NUMBER(2)

FACULTY
Name              Null?          Type
-------------- -------- ------------------------------
 FACULTY_ID       NOT NULL       NUMBER(2)
 FACULTY_NAME                    VARCHAR2(20)
 LOCATION_ID                     NUMBER(2)
```

Examine the following two SQL statements:

Statement 1

```
SQL>SELECT faculty_name,COUNT(student_id)
FROM student JOIN faculty
USING (faculty_id, location_id)
GROUP BY faculty_name;
```

Statement 2

```
SQL>SELECT faculty_name,COUNT(student_id)
FROM student NATURAL JOIN faculty
GROUP BY faculty_name;
```

Which statement is true regarding the outcome?

A. Only s tatement 1 executes successfully and gives the required result.
B. Only statement 2 executes successfully and gives the required result.
C. Both statements 1 and 2 execute successfully and give different results.
D. Both statements 1 and 2 execute successfully and give the same required result.

Answer: D
Section: (none)

Explanation/Reference:

QUESTION 178
View the Exhibits and examine the structures of the PRODUCTS, SALES, and CUSTOMERS tables.

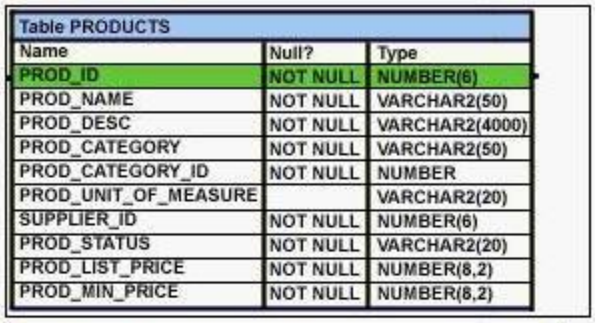


Table PRODUCTS

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER(6)
PROD_NAME	NOT NULL	VARCHAR2(50)
PROD_DESC	NOT NULL	VARCHAR2(4000)
PROD_CATEGORY	NOT NULL	VARCHAR2(50)
PROD_CATEGORY_ID	NOT NULL	NUMBER
PROD_UNIT_OF_MEASURE		VARCHAR2(20)
SUPPLIER_ID	NOT NULL	NUMBER(6)
PROD_STATUS	NOT NULL	VARCHAR2(20)
PROD_LIST_PRICE	NOT NULL	NUMBER(8,2)
PROD_MIN_PRICE	NOT NULL	NUMBER(8,2)

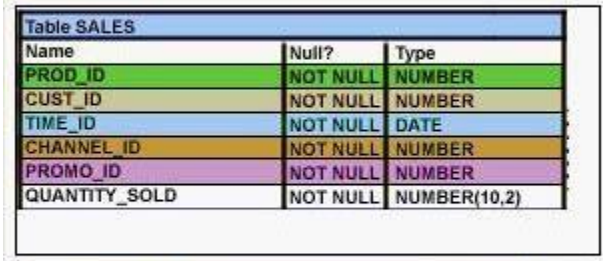


Table SALES

Name	Null?	Type
PROD_ID	NOT NULL	NUMBER
CUST_ID	NOT NULL	NUMBER
TIME_ID	NOT NULL	DATE
CHANNEL_ID	NOT NULL	NUMBER
PROMO_ID	NOT NULL	NUMBER
QUANTITY_SOLD	NOT NULL	NUMBER(10,2)

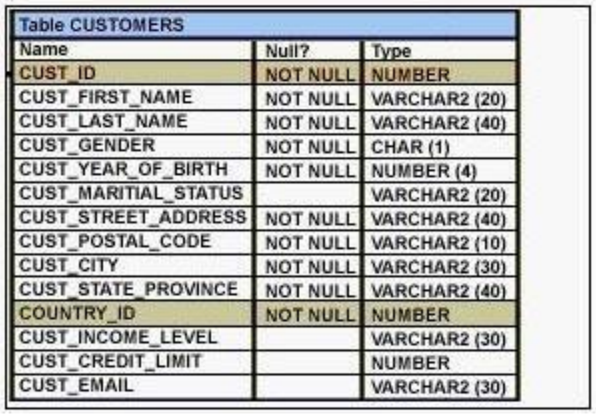


Table CUSTOMERS

Name	Null?	Type
CUST_ID	NOT NULL	NUMBER
CUST_FIRST_NAME	NOT NULL	VARCHAR2 (20)
CUST_LAST_NAME	NOT NULL	VARCHAR2 (40)
CUST_GENDER	NOT NULL	CHAR (1)
CUST_YEAR_OF_BIRTH	NOT NULL	NUMBER (4)
CUST_MARITIAL_STATUS		VARCHAR2 (20)
CUST_STREET_ADDRESS	NOT NULL	VARCHAR2 (40)
CUST_POSTAL_CODE	NOT NULL	VARCHAR2 (10)
CUST_CITY	NOT NULL	VARCHAR2 (30)
CUST_STATE_PROVINCE	NOT NULL	VARCHAR2 (40)
COUNTRY_ID	NOT NULL	NUMBER
CUST_INCOME_LEVEL		VARCHAR2 (30)
CUST_CREDIT_LIMIT		NUMBER
CUST_EMAIL		VARCHAR2 (30)

You need to generate a report that gives details of the customer's last name, name of the product, and the quantity sold for all customers in ' Tokyo' .

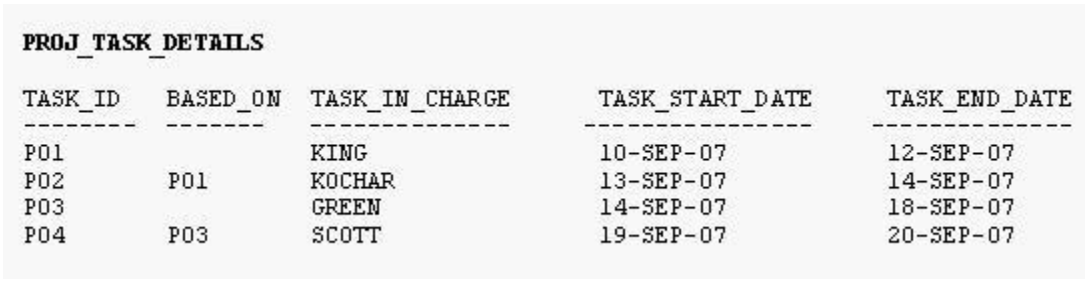
Which two queries give the required result? (Choose two.)

A. SELECT c.cust_last_name,p.prod_name, s.quantity_sold
   FROM sales s JOIN products p
   USING(prod_id)
   JOIN customers c
   USING(cust_id)
   WHERE c.cust_city='Tokyo';
B. SELECT c.cust_last_name, p.prod_name, s.quantity_sold
   FROM products p JOIN sales s JOIN customers c
   ON(p.prod_id=s.prod_id)
   ON(s.cust_id=c.cust_id)
   WHERE c.cust_city='Tokyo';
C. SELECT c.cust_last_name, p.prod_name, s.quantity_sold
   FROM products p JOIN sales s
   ON(p.prod_id=s.prod_id)
   JOIN customers c
   ON(s.cust_id=c.cust_id)
   AND c.cust_city='Tokyo';
D. SELECT c.cust_id,c.cust_last_name,p.prod_id, p.prod_name, s.quantity_sold FROM products p JOIN sales s
   USING(prod_id)
   JOIN customers c
   USING(cust_id)
   WHERE c.cust_city='Tokyo';

Answer: AC
Section: (none)

Explanation/Reference:

QUESTION 179
View the Exhibit and examine the data in the



```
PROJ_TASK_DETAILS

TASK_ID   BASED_ON  TASK_IN_CHARGE     TASK_START_DATE    TASK_END_DATE
--------  -------   --------------   ----------------   --------------
P01                 KING               10-SEP-07          12-SEP-07
P02       P01       KOCHAR             13-SEP-07          14-SEP-07
P03                 GREEN              14-SEP-07          18-SEP-07
P04       P03       SCOTT              19-SEP-07          20-SEP-07
```

The PROJ_TASK_DETAILS table stores information about tasks involved in a project and the relation between them. The BASED_ON column indicates dependencies between tasks. Some tasks do not depend on the completion of any other tasks.

You need to generate a report showing all task IDs, the corresponding task ID they are dependent on, and the name of the employee in charge of the task it depends on.

Which query would give the required result?

A. SELECT p.task_id, p.based_on, d.task_in_charge
   FROM proj_task_details p JOIN proj_task_details d
   ON (p.based_on = d.task_id);
B. SELECT p.task_id, p.based_on, d.task_in_charge
   FROM proj_task_details p LEFT OUTER JOIN proj_task_details d ON (p.based_on = d.task_id);
C. SELECT p.task_id, p.based_on, d.task_in_charge
   FROM proj_task_details p FULL OUTER JOIN proj_task_details d ON (p.based_on = d.task_id);
D. SELECT p.task_id, p.based_on, d.task_in_charge
   FROM proj_task_details p JOIN proj_task_details d

```
    ON (p.task_id = d.task_id);
```

Answer: B
Section: (none)

Explanation/Reference:

QUESTION 180
Which two statements are true regarding subqueries? (Choose two.)

A. A subquery can retrieve zero or more rows.
B. Only two subqueries can be placed at one level.
C. A subquery can be used only in SQL query statements.
D. A subquery can appear on either side of a comparison operator.
E. There is no limit on the number of subquery levels in the WHERE clause of a SELECT statement.

Answer: AD
Section: (none)

Explanation/Reference:
Define Subqueries
A subquery is a query that is nested inside a SELECT, INSERT, UPDATE, or DELETE statement or inside another subquery. A subquery can return a set of rows or just one row to its parent query. A scalar subquery is a query that returns exactly one value: a single row, with a single column. Scalar subqueries can be used in most places in a SQL statement where you could use an expression or a literal value.
The places in a query where a subquery may be used are as follows:
■ In the SELECT list used for column projection
■ In the FROM clause
■ In the WHERE clause
■ In the HAVING clause

Subqueries can be nested to an unlimited depth in a FROM clause but to “only” 255 levels in a WHERE clause. They can be used in the SELECT list and in the FROM, WHERE, and HAVING clauses of a query.