

Question Results

Score 1 of 1

Question:

Review the first two illustrations as well as the `ONLINE_SUBSCRIBERS` table and then review this SQL code:

```
SELECT * FROM FURNISHING:
```

CAT#	ITEM_NAME	ADDED	SECTION
-----	-----	-----	-----
1	Side table	23-DEC-09	LR
2	Desk	12-SEP-09	BR
3	Towel	10-OCT-09	BA

```
SELECT * FROM STORE_INVENTORY:
```

NUM	AISLE	PRODUCT	LAST_ORDER
-----	-----	-----	-----
77	F02	Jacket	2009-09-09
78	B11	Towel	2009-11-11
79	SP01	Lava lamp	2009-12-21

FURNISHINGS	
P * CAT#	NUMBER
ITEM_NAME	VARCHAR2 (15 BYTE)
ADDED	DATE
SECTION	VARCHAR2 (10 BYTE)
🔑 PK_CAT#	

STORE_INVENTORY	
P * NUM	NUMBER
aisle	VARCHAR2 (7 BYTE)
PRODUCT	VARCHAR2 (15 BYTE)
LAST_ORDER	DATE
🔑 PK_NUM	

ONLINE_SUBSCRIBERS	
P * ONLINE_SUBSCRIBER_ID	NUMBER
SUB_DATE	DATE
EMAIL	VARCHAR2 (120 BYTE)
FIRSTNAME	VARCHAR2 (20 BYTE)
LASTNAME	VARCHAR2 (30 BYTE)
COMPANY	VARCHAR2 (30 BYTE)
🔑 PK_ONLINE_SUBSCRIBER_ID	

```
SELECT ONLINE_SUBSCRIBER_ID, EMAIL
FROM ONLINE_SUBSCRIBERS;
```


ONLINE_SUBSCRIBER_ID	EMAIL
1	pendicott77@kasteelinc.com
2	watcher@foursigma.org
3	hardingpal@ckofca.com

```
01 SELECT A.SUB_DATE, COUNT(*)
02 FROM ONLINE_SUBSCRIBERS A JOIN
03 (SELECT LAST_ORDER, PRODUCT FROM STORE_INVENTORY
04 UNION
05 SELECT ADDED, ITEM_NAME FROM FURNISHINGS) B
06 ON A.SUB_DATE = B.LAST_ORDER
07 GROUP BY A.SUB_DATE;
```


Where can you add an ORDER BY to this code?

(Choose two.)

Response:

 At the end of line 5 before the right parenthesis

Nowhere

 After line 7

Between lines 5 and 6

Score 1 of 1

Question:


Which statement is true about an inner join specified in the WHERE clause of a query?

Response:

It must have primary-key and foreign-key constraints defined on the columns used in the join condition.

It is applicable for only equijoin conditions.

It requires the column names to be the same in all tables used for the join conditions.

 It is applicable for equijoin and nonequijoin conditions.

Score 1 of 1

Question:

Which of the following forms of subquery never returns more than one row?

Response:

Multiple-column

None of the above



Scalar

Correlated

Score 0 of 1

Question:

Which two statements are true about Data Manipulation Language (DML) statements?

Response:



A DELETE FROM statement can remove rows based on only a single condition on a table.



AN INSERT INTO. . .VALUES. . statement can add multiple rows per execution to a table.



An UPDATE...SET... statement can modify multiple rows based on multiple conditions on a table.

An INSERT INTO...VALUES..... statement can add a single row based on multiple conditions on a table.

An UPDATE...SET.... statement can modify multiple rows based on only a single condition on a table.



A DELETE FROM..... statement can remove multiple rows based on multiple conditions on a table.

Score 1 of 1

Question:

Which of the following is true of character functions?

Response:

They always return a character value.



They are generally used to process text data.

They always accept characters as parameters and nothing else.

They generally have the letters CHAR somewhere in the function name.

Score 1 of 1

Question:

Examine the following two claims:

[1] The DBA_TAB_PRIVS data dictionary view allows a user account to see object privileges it has granted to other user accounts.

[2] The DBA_TAB_PRIVS data dictionary view allows a user account to see object privileges granted by other user accounts to itself.

Which of these claims is true?

Response:



Both 1 and 2

Only 2

Only 1

Neither 1 nor 2

Score 1 of 1

Question:

You want to display 5 percent of the rows from the sales table for products with the lowestAMOUNT_SOLD and also want to include the rows that have the sameAMOUNT_SOLD even if this causes the output to exceed 5 percent of the rows.

Which query will provide the required result?

Response:

```
SELECT prod_id, cust_id, amount_sold FROM sales  
ORDER BY amount_sold  
FETCH FIRST 5 PERCENT ROWS ONLY WITH TIES;
```

```
SELECT prod_id, cust_id, amount_sold FROM sales  
ORDER BY amount sold  
FETCH FIRST 5 PERCENT ROWS ONLY;
```

```
SELECT prod_ id, cust_id, amount_sold FROM sales
```


ORDER BY amount_sold
FETCH FIRST 5 PERCENT ROWS WITH TIES ONLY;
✓ SELECT prod_id, cust_id, amount_sold FROM sales
ORDER BY amount_sold
FETCH FIRST 5 PERCENT ROWS WITH TIES;

Score 1 of 1

Question:

Examine the structure of the members table:

Name	Null?	Type
MEMBER_ID	NOT NULL	VARCHAR2 (6)
FIRST_NAME		VARCHAR2 (50)
LAST_NAME	NOT NULL	VARCHAR2 (50)
ADDRESS		VARCHAR2 (50)
CITY		VARCHAR2 (25)
STATE		VARCHAR2 (3)

You want to display details of all members who reside in states starting with the letter A followed by exactly one character. Which SQL statement must you execute?

Response:

SELECT * FROM MEMBERS WHERE state LIKE '%A_' ;

SELECT * FROM MEMBERS WHERE state LIKE 'A_%';

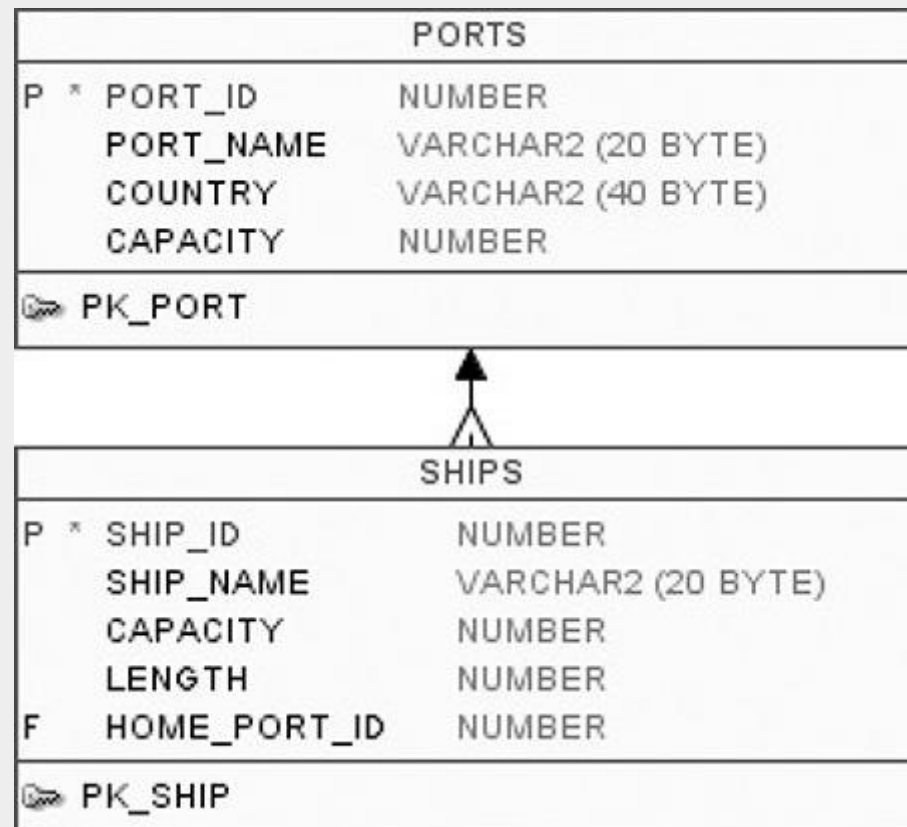
SELECT * FROM MEMBERS WHERE state LIKE 'A%';

✓ SELECT * FROM MEMBERS WHERE state LIKE 'A_';

Score 1 of 1

Question:

Review the illustration and the following SQL code:



```
01 UPDATE PORTS P
02 SET     CAPACITY = CAPACITY + 1
03 WHERE   EXISTS (SELECT *
04                      FROM SHIPS
05                      WHERE HOME_PORT_ID = P.PORT_ID) ;
```

The PORTS table has 15 rows. The SHIPS table has 20 rows. Each row in PORTS has a unique value for PORT_ID. Each PORT_ID value is represented in the HOME_PORT_ID column of at least one row of the SHIPS table.

What can be said of this UPDATE statement?

Response:



The value for CAPACITY will increase once for each of the 15 rows in the PORTS table.

The value for CAPACITY will increase by 20 for each of the 15 rows in the PORTS table.

The value for CAPACITY will not increase.

The statement will fail to execute because of an error in the syntax.

Score 1 of 1

Question:

Review the first two illustrations and then review this SQL code:

```
SELECT * FROM FURNISHING:
```

CAT#	ITEM_NAME	ADDED	SECTION
-----	-----	-----	-----
1	Side table	23-DEC-09	LR
2	Desk	12-SEP-09	BR
3	Towel	10-OCT-09	BA

```
SELECT * FROM STORE_INVENTORY:
```

NUM	AISLE	PRODUCT	LAST_ORDER
-----	-----	-----	-----
77	F02	Jacket	2009-09-09
78	B11	Towel	2009-11-11
79	SP01	Lava lamp	2009-12-21

FURNISHINGS	
P * CAT#	NUMBER
ITEM_NAME	VARCHAR2 (15 BYTE)
ADDED	DATE
SECTION	VARCHAR2 (10 BYTE)
🔑 PK_CAT#	

STORE_INVENTORY	
P * NUM	NUMBER
aisle	VARCHAR2 (7 BYTE)
PRODUCT	VARCHAR2 (15 BYTE)
LAST_ORDER	DATE
🔑 PK_NUM	

```

01  SELECT '---', SECTION
02  FROM    FURNISHINGS
03  WHERE   CAT# NOT IN (1,2)
04  UNION ALL
05  SELECT TO_CHAR(LAST_ORDER, 'Month'), aisle
06  FROM    STORE_INVENTORY;

```

How many rows will result from this query?

Response:

It will not execute because it will fail with a syntax error.

0



4

6

Score 1 of 1

Question:

Which three statements are true regarding the data types?

Response:



Only one LONG column can be used per table.



The value for a CHAR data type column is blank-padded to the maximum defined column width.



The minimum column width that can be specified for a varchar2 data type column is one.

ATIMESTAMP data type column stores only time values with fractional seconds.

The BLOB data type column is used to store binary data in an operating system file.

Score 1 of 1

Question:

Examine the structure of the **BOOKS_TRANSACTIONS** table:

Name	Null?	Type
TRANSACTION_ID	NOT NULL	VARCHAR2 (6)
BORROWED_DATE		VARCHAR2 (50)
DUE_DATE		DATE
BOOK_ID		DATE
MEMBER_ID		VARCHAR2 (6)

You want to display the member IDs, due date, and late fee as \$2 for all transactions. Which SQL statement must you execute?

Response:

```
SELECT member_id 'MEMBER ID', due_date 'DUE DATE', '$2 AS LATE FEE' FROM  
BOOKS_TRANSACTIONS;
```



```
SELECT member_id AS "MEMBER ID", due_date AS "DUE DATE", '$2' AS "LATE FEE"  
FROM BOOKS_TRANSACTIONS;
```

```
SELECT member_id AS MEMBER_ID, due_date AS DUE_DATE, $2 AS LATE_FEE FROM  
BOOKS_TRANSACTIONS;
```

```
SELECT member_id AS "MEMBER ID", due_date AS "DUE DATE", $2 AS "LATE FEE"  
FROM BOOKS_TRANSACTIONS;
```

Score 0 of 1

Question:

Examine the commands used to create DEPARTMENT_DETAILS and COURSE_DETAILS:

```
SQL> CREATE TABLE DEPARTMENT_DETAILS
(DEPARTMENT_ID    NUMBER    PRIMARY KEY ,
 DEPARTMENT_NAME  VARCHAR2(50) ,
 HOD              VARCHAR2(50));
SQL> CREATE TABLE COURSE_DETAILS
(COURSE_ID        NUMBER    PRIMARY KEY ,
 COURSE_NAME      VARCHAR2 (50) ,
 DEPARTMENT_ID    NUMBER    REFERENCES DEPARTMENT_DETAILS (DEPARTMENT_ID));
```

You want to generate a list of all department IDs along with any course IDs that may have been assigned to them. Which SQL statement must you use?

A)
Exhibit

```
SELECT d.department_id, c.course_id FROM department_details d RIGHT OUTER JOIN
course_details c ON (d.department_id=c.department_id);
```

B)
Exhibit

```
SELECT d.department_id, c.course_id FROM department_details d LEFT OUTER JOIN
course_details c ON (d.department_id=c.department_id);
```

C)
Exhibit


```
SELECT d.department_id, c.course_id FROM course_details c LEFT OUTER JOIN
department_details d ON (c.department_id=d.department_id);
```

D)
Exhibit


```
SELECT d.department_id, c.course_id FROM department_details d RIGHT OUTER JOIN
course_details c ON (c.department_id=d.department_id);
```


Response:

Option A

 Option D

Option C


 Option B

Score 1 of 1

Question:

The exam is timed.

Response:


 True

False

Score 1 of 1

Question:

Review the illustration and then look at the SQL code that follows:

PROJECTS		
P *	PROJECT_ID	NUMBER
	SHIP_ID	NUMBER
	PURPOSE	VARCHAR2 (30 BYTE)
	PROJECT_NAME	VARCHAR2 (40 BYTE)
	PROJECT_COST	NUMBER
	DAYS	NUMBER
 PK_PROJECT_ID		

```

01  SELECT  COUNT (COUNT (PROJECT_COST) )
02  FROM    PROJECTS
03  GROUP BY PURPOSE;

```


What will happen if you try to execute this query on the **PROJECTS** table?

Response:

It will fail with a syntax error because line 1 is not correct.

It will fail with an execution error because you cannot use a VARCHAR2 column in a GROUP BY clause.

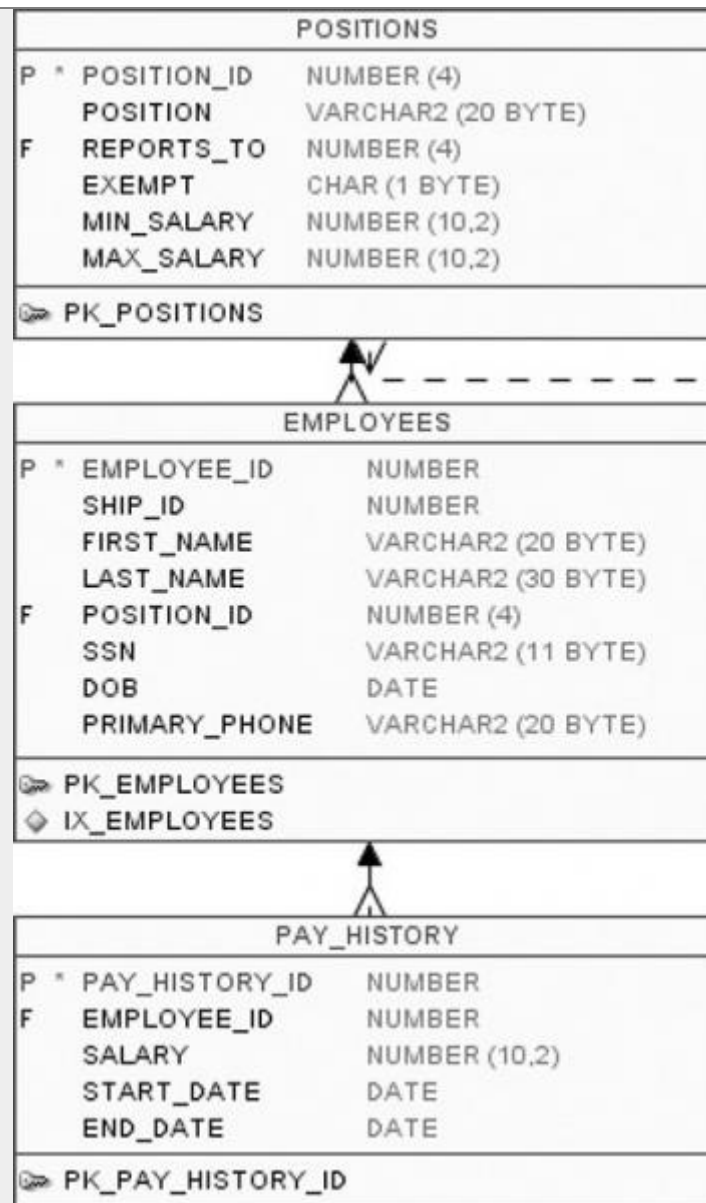
It will succeed and display one row for each different value in the PURPOSE column.

 It will succeed and display one row.

Score 1 of 1

Question:

Review the illustration. Which of the following is a valid self-join statement?
(Choose all that apply.)



Response:

✓
SELECT P1.POSITION_ID, P1.MIN_SALARY, P1.MAX_SALARY
FROM POSITIONS P1 INNER JOIN POSITIONS P2
ON P1.REPORTS_TO = P2.POSITION_ID;

✓
SELECT P1.POSITION_ID, P1.MIN_SALARY, P1.MAX_SALARY
FROM POSITIONS P1 RIGHT OUTER JOIN POSITIONS P2
ON P1.REPORTS_TO = P2.POSITION_ID;

✓
SELECT P1.POSITION_ID, P1.MIN_SALARY, P1.MAX_SALARY
FROM POSITIONS P1 JOIN POSITIONS P2
ON P1.REPORTS_TO = P2.POSITION_ID;

SELECT P1.POSITION_ID, P1.MIN_SALARY, P1.MAX_SALARY
FROM POSITIONS P1 SELF JOIN POSITIONS P2
ON P1.REPORTS_TO = P2.POSITION_ID;

Score 0 of 1

Question:


Which of the following aggregate functions can be used on character data?
(Choose two.)

Response:

AVG

 COUNT

 MEDIAN

 MIN


Score 1 of 1


Question:

Which two statements are true regarding the SQL GROUP BY clause?

Response:

You can use a column alias in the GROUP BY clause.

 Using the WHERE clause before the GROUP BY clause excludes rows before creating groups.

 if the SELECT clause has an aggregating function, then columns without an aggregating function in the SELECT clause should be included in the GROUP BY clause.

Using the WHERE clause after the GROUP BY clause excludes rows after creating groups.

The GROUP BY clause is mandatory if you are using an aggregating function in the SELECT clause.


Score 1 of 1

Question:

Review the illustration. Your assignment: create a SELECT statement that queries the PROJECTS table to show the average project cost for each PURPOSE.

You know there are only two values for PURPOSE in the table: 'Upgrade' and 'Maintenance'. You want to restrict output to those rows where DAYS is greater than 3.

Which of the following SELECT statements will perform this task?

PROJECTS		
P *	PROJECT_ID	NUMBER
	SHIP_ID	NUMBER
	PURPOSE	VARCHAR2 (30 BYTE)
	PROJECT_NAME	VARCHAR2 (40 BYTE)
	PROJECT_COST	NUMBER
	DAYS	NUMBER
 PK_PROJECT_ID		

Response:

```
SELECT PURPOSE, AVG (PROJECT_COST)
FROM PROJECTS
GROUP BY PURPOSE, (DAYS > 3);
```

```
SELECT PURPOSE, AVG (PROJECT_COST)
FROM PROJECTS
WHERE DAYS > 3
GROUP BY PURPOSE, DAYS
HAVING DAYS > 3;
```



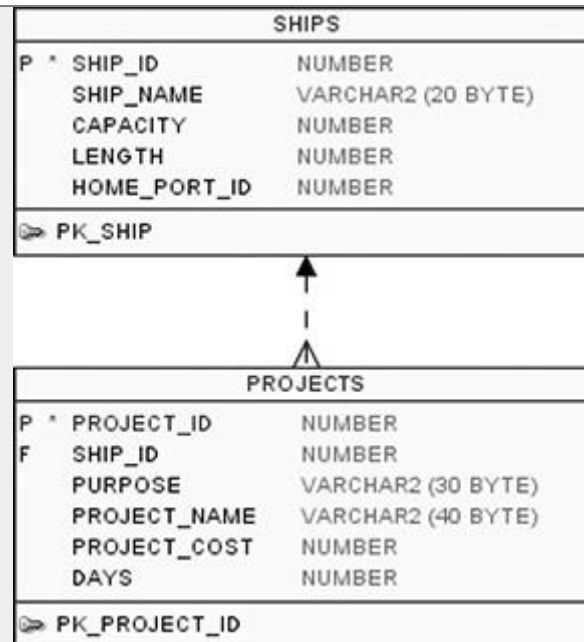
```
SELECT PURPOSE, AVG (PROJECT_COST)
FROM PROJECTS
WHERE DAYS > 3
GROUP BY PURPOSE;
```

```
SELECT PURPOSE, AVG (PROJECT_COST)
FROM PROJECTS
GROUP BY PURPOSE
HAVING DAYS > 3;
```

Score 1 of 1

Question:

Review the following illustration:



Now review the following SQL code:

```


01 CREATE OR REPLACE VIEW SHIP_CAP_PROJ AS
02 SELECT    SHIP_ID,
03           TO_CHAR(CAPACITY, '999,999'),
04           PROJECT_COST
05 FROM      SHIPS JOIN PROJECTS
06 USING    (SHIP_ID)
07 WHERE    (PROJECT_COST * 2) < 100000;
  
```

What will result from an attempt to execute this SQL code?

Response:

The statement will fail because of an error in line 6.

The statement will fail because of an error in line 7.

 The statement will fail because of an error in line 3.

The statement will execute, and the view will be successfully created.

Score 1 of 1

Question:

Review the first two illustrations and then review this SQL code

```
SELECT * FROM FURNISHING:
```

CAT#	ITEM_NAME	ADDED	SECTION
-----	-----	-----	-----
1	Side table	23-DEC-09	LR
2	Desk	12-SEP-09	BR
3	Towel	10-OCT-09	BA

```
SELECT * FROM STORE_INVENTORY:
```

NUM	AISLE	PRODUCT	LAST_ORDER
-----	-----	-----	-----
77	F02	Jacket	2009-09-09
78	B11	Towel	2009-11-11
79	SP01	Lava lamp	2009-12-21

FURNISHINGS	
P * CAT#	NUMBER
ITEM_NAME	VARCHAR2 (15 BYTE)
ADDED	DATE
SECTION	VARCHAR2 (10 BYTE)
🔑 PK_CAT#	

STORE_INVENTORY	
P * NUM	NUMBER
aisle	VARCHAR2 (7 BYTE)
PRODUCT	VARCHAR2 (15 BYTE)
LAST_ORDER	DATE
🔑 PK_NUM	

```
SELECT NUM, PRODUCT FROM STORE_INVENTORY
INTERSECT
SELECT CAT#, ITEM_NAME FROM FURNISHINGS;
```

How many rows will result from this query?

Response:

6

1

3

 0

Score 0 of 1

Question:

The difference between dropping a column from a table with DROP and setting a column to be UNUSED is:

Response:



The UNUSED column and its data are retained within the table's storage allocation and counts against the total limit on the number of columns the table is allowed to have.

Nothing.



A column that is dropped with DROP no longer appears within the table's description as shown with the DESC or DESCRIBE statement, whereas a column that is set to UNUSED still appears in the table's structure as shown in the output of the DESC statement.

An UNUSED column can be recovered.

Score 1 of 1

Question:

Consider the following set of SQL statements:

```
CREATE TABLE INSTRUCTORS
(INSTRUCTOR_ID NUMBER,
NAME          VARCHAR2(20),
CONSTRAINT   ID_PK  PRIMARY KEY (INSTRUCTOR_ID),
CONSTRAINT   NAME_UN UNIQUE (NAME));


INSERT INTO INSTRUCTORS (INSTRUCTOR_ID, NAME)
VALUES (1, 'Howard Jackson');
INSERT INTO INSTRUCTORS (INSTRUCTOR_ID, NAME)
VALUES (2, 'Trish Mars');
```

The table will create successfully. What will be the result of the two INSERT statements?

Response:

The first will fail, but the second will execute.

Neither will execute.

 Both will execute successfully.

The first will execute, but the second will fail.

Score 1 of 1

Question:

The difference between an INNER and an OUTER join is:

Response:

The INNER runs on data inside the table; the OUTER runs on data outside of the table.

The INNER join relates a table to itself; the OUTER join relates a table to other tables.

The OUTER join relates a table to tables in other user accounts; the INNER does not.



The INNER join displays rows that match in all joined tables; the OUTER join shows data that doesn't necessarily match.

Score 1 of 1

Question:

You are logged in to user FINANCE. It is currently the only schema in the entire database. The following exist in the database:

- A VIEW named VENDORS
- A CONSTRAINT named VENDORS
- An INDEX named CUSTOMER#ADDRESS

You attempt to execute the following SQL statement:

```
CREATE TABLE CUSTOMER#ADDRESS  
(ID NUMBER,  
  NAME VARCHAR2(30));
```

Which one of the following is true?

Response:

The question is flawed because you cannot have an INDEX named CUSTOMER#ADDRESS.



The SQL statement will execute, and the TABLE will be created.

The question is flawed because you cannot have a VIEW and a CONSTRAINT with identical names in the same schema.

The SQL statement will fail to execute and result in an error message because you cannot create a TABLE that has the same name as an INDEX in the same schema.

The SQL statement will fail to execute and result in an error message because you cannot create a TABLE name with the # character.

Score 1 of 1

Question:

Which of the following data dictionary views does not have an OWNER column?

Response:



USER_TABLES

ALL_INDEXES

DBA_CONS_COLUMNS

All of the above

Score 1 of 1

Question:

An aggregate function can be called from within:
(Choose two.)

Response:



The select list of a SELECT statement

The expression list of a DELETE statement

The HAVING clause of an INSERT statement



The ORDER BY clause of a SELECT statement

Score 0 of 1

Question:

Which of the following keywords cannot be used with the CREATE SEQUENCE statement?

Response:



MAXVALUE



JOIN


CYCLE

INCREMENT

Score 1 of 1

Question:

Review the illustration and then look at the SQL code that follows:

CRUISE_ORDERS	
P * CRUISE_ORDER_ID	NUMBER
P * ORDER_DATE	DATE
 PK_CO	

```
01 SELECT TO_CHAR(ORDER_DATE,'Q') "Quarter", COUNT(*)
02 FROM CRUISE_ORDERS
03 WHERE TO_CHAR(ORDER_DATE,'YYYY') = '2009'
04 GROUP BY TO_CHAR(ORDER_DATE,'Q');
```

Recall that the 'Q' format model is for quarter, so TO_CHAR using a DATE data type with the 'Q' format mask is translating the date into the quarter in which it falls—1, 2, 3, or 4.

Given that, which of the following statements is true of the SQL statement?

Response:

None of the above.

It will fail because of a syntax error in line 1 since you cannot use the TO_CHAR function with the COUNT aggregate function.



It will execute and show the number of orders in the CRUISE_ORDERS table for each quarter in the year 2009.

It will fail because of a syntax error in line 4 since you cannot use the TO_CHAR function in the GROUP BY clause.

Score 1 of 1

Question:

The CASCADE keyword, when used with TRUNCATE:

Response:

Can be used with the optional DEPENDENCY keyword

Is required if the table has any dependent child tables



None of the above

Will ensure that future attempts to insert rows to the table will be rejected if they satisfy the TRUNCATE table's WHERE clause

Score 1 of 1

Question:

Which two statements are true regarding roles?


(Choose two.)

Response:

A user can be granted only one role at any point of time.

A role can be granted to itself.

The REVOKE command can be used to remove privileges but not roles from other users.

 Roles are named groups of related privileges that can be granted to users or other roles.

 A role can be granted to PUBLIC.

Score 1 of 1

Question:

Review this SQL statement: `SELECT MONTHS_BETWEEN(LAST_DAY('15-JAN-12')+1,'01-APR-12')`FROM DUAL; What will result from this query?

Response:

> 2 (some number greater than 2)



-2

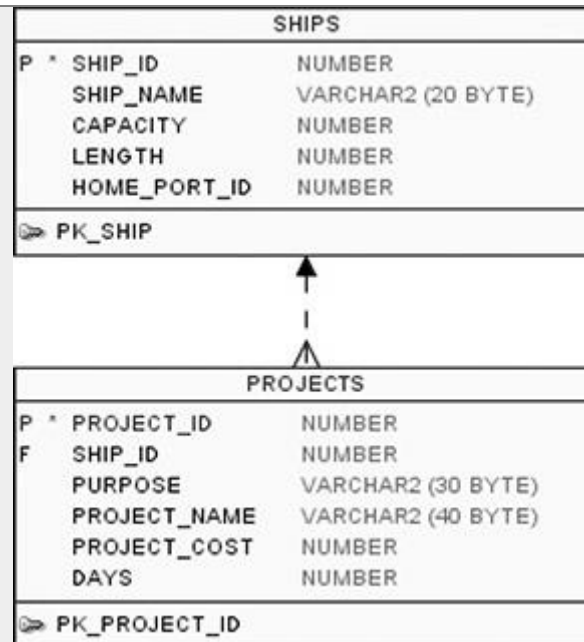
2

< -2 (some number less than negative 2)

Score 1 of 1

Question:

Review the illustration and the following SQL code:



```

01 CREATE OR REPLACE VIEW MAJOR_PROJECTS AS
02     SELECT PROJECT_ID, SHIP_ID, PROJECT_NAME, PROJECT_COST
03     FROM   PROJECTS
04     WHERE  PROJECT_COST > 10000;
05
06 INSERT INTO MAJOR_PROJECTS
07     (PROJECT_ID, SHIP_ID, PROJECT_NAME, PROJECT_COST)
08     VALUES
09     ((SELECT MAX(PROJECT_ID)+1 FROM PROJECTS),
10     (SELECT MAX(SHIP_ID) FROM SHIPS),
11     'Small Project',
12     500);
  
```

What will result from an attempt to execute these two SQL statements?

Response:

The INSERT statement will fail because the PROJECT_COST value being inserted is not consistent with the WHERE clause on line 4.

The CREATE statement will fail because it omits the PURPOSE column from the PROJECTS table.

The INSERT statement will fail because of an error on lines 9 and 10.




The CREATE and INSERT statements will successfully execute.

Score 1 of 1

Question:

Review the following illustration:

CRUISE_ORDERS	
P * CRUISE_ORDER_ID	NUMBER
P * ORDER_DATE	DATE
 PK_CO	

Now review this SQL statement:

```
SELECT CRUISE_ORDER_ID, COUNT(ORDER_DATE)
FROM   CRUISE_ORDERS;
```

What can be said of this statement?

Response:

It will fail to execute because ORDER_DATE is a date data type, and no aggregate function can work with a date data type.

There is nothing wrong with the SQL statement.



It will fail to execute because it mixes scalar and aggregate data in the select list.

It will execute successfully but not produce any meaningful output.

Score 1 of 1

Question:

View the exhibit and examine the structures of the EMPLOYEES and DEPARTMENTS tables.

EMPLOYEES

Name Null? Type

EMPLOYEE_ID NOT NULL NUMBER(6)

FIRST_NAME VARCHAR2(20)

LAST_NAME NOT NULL VARCHAR2(25)

HIRE_DATE NOT NULL DATE

JOB_ID NOT NULL VARCHAR2(10)

SALARY NUMBER(10,2)

COMMISSION NUMBER(6,2)

MANAGER_ID NUMBER(6)

DEPARTMENT_ID NUMBER(4)

DEPARTMENTS

Name Null? Type

DEPARTMENT_ID NOT NULL NUMBER(4)
DEPARTMENT_NAME NOT NULL VARCHAR2(30)
MANAGER_ID NUMBER(6)
LOCATION_ID NUMBER(4)

You want to update EMPLOYEES table as follows:

- 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 outcome?

Response:



It executes successfully but does not give the correct result.

It generates an error because a subquery cannot have a join condition in a UPDATE statement.

It executes successfully and gives the correct result.

It generates an error because multiple columns (SALARY, COMMISSION) cannot be specified together in an UPDATE statement.

Score 1 of 1

Question:

Which subquery includes references to the parent query and thus cannot execute as a standalone query?
(Choose the best answer.)

Response:



A correlated subquery

A multiple-column subquery

A referential subquery

A scalar subquery