Question Text: Relation R(x,y) currently consists of only the one tuple (NULL, NULL). Which of the following SQL-99 standard queries will produce a nonempty output? That is, at least one tuple will be produced, although the tuple(s) may have NULL's.

Correct Choice 1: SELECT * FROM R WHERE x IS NULL

Choice Explanation: **NONE**

Correct Choice 2: SELECT * FROM R WHERE y IS NULL

Choice Explanation: **NONE**

Correct Choice 3: SELECT * FROM R

Choice Explanation: **NONE**

Correct Choice 4: SELECT x FROM R WHERE EXISTS (SELECT y FROM R)

Choice Explanation: **NONE**

Correct Choice 5: SELECT y FROM R WHERE EXISTS (SELECT x FROM R)

Choice Explanation: **NONE**

Question Explanation: There are several things about NULL's to remember. First, the proper way to ask whether an attribute like x has the value NULL in some tuple is "x IS NULL" (or "x IS NOT NULL" for the opposite). It is not correct, standard SQL to say "x = NULL".

Second, remember that in SQL, the truth value of a condition like x=y or x=1, when x has the value NULL is UNKNOWN, not FALSE. Moreover, for a WHERE-condition to be satisfied, the truth value must be TRUE; UNKNOWN is not good enough. For example, even if (x,y) is the tuple (NULL, NULL), the condition "WHERE x=y" is not TRUE for this tuple, and (NULL, NULL) would not be passed to the output.

The correct choices for this question include some, like SELECT * FROM R WHERE x IS NULL, that properly ask whether one of the components of the tuple (NULL, NULL) is in fact NULL, and thus yield something to be output. Note that the value of "x IS NULL" is TRUE, not UNKNOWN. Several other choices, like SELECT x FROM R WHERE EXISTS (SELECT y FROM R), exploit the idea that even the tuple (NULL) is enough to make an output nonempty, and therefore to satisfy an EXISTS-condition.

Question Text:

The table
Scores(Team, Day, Opponent, Runs)
Gives the scores in the Japanese Baseball League for two consecutive days. The Opponent is NULL if the Team did not play on that day. The number of Runs is given as NULL if either the team did not play, or will play on that day but the game is not yet concluded. The data in this table is as follows:

Team	Day	Opponent	Runs
Dragons	Sunday	Swallows	4
Tigers	Sunday	Bay Stars	9
Carp	Sunday	NULL	NULL
Swallows	Sunday	Dragons	7
Bay Stars	Sunday	Tigers	2
Giants	Sunday	NULL	NULL
Dragons	Monday	Carp	NULL
Tigers	Monday	NULL	NULL
Carp	Monday	Dragons	NULL
Swallows	Monday	Giants	0
Bay Stars	Monday	NULL	NULL
Giants	Monday	Swallows	5

What is the result of executing on this data the query:

```
  SELECT Team, Day
  FROM Scores
  WHERE Opponent IS NULL OR
    NOT (Runs >= 0)
```

Question Explanation: The first condition in the WHERE clause asks for all rows where the Opponent column is NULL (rows 3, 6, 8, and 11) or the Runs column is at least 0. Even though we know that it is impossible to score a negative number of runs, the truth value of "Runs>=NULL" is "unknown". Thus, the truth value of "NOT Runs>=NULL" is also "unknown", and the second condition of the WHERE clause is not "true" for any of the rows. Thus, only the four rows with NULL in the Opponent column meet the condition. The resulting output is:

Team	Day
Carp	Sunday
Giants	Sunday
Tigers	Monday

Bay Stars	Monday

Question Text: The table

Scores(Team, Day, Opponent, Runs)

Gives the scores in the Japanese Baseball League for two consecutive days. The Opponent is NULL if the Team did not play on that day. The number of Runs is given as NULL if either the team did not play, or will play on that day but the game is not yet concluded. The data in this table is as follows:

Team	Day	Opponent	Runs
Dragons	Sunday	Swallows	4
Tigers	Sunday	Bay Stars	9
Carp	Sunday	NULL	NULL
Swallows	Sunday	Dragons	7
Bay Stars	Sunday	Tigers	2
Giants	Sunday	NULL	NULL
Dragons	Monday	Carp	NULL
Tigers	Monday	NULL	NULL
Carp	Monday	Dragons	NULL
Swallows	Monday	Giants	0
Bay Stars	Monday	NULL	NULL
Giants	Monday	Swallows	5

Consider the following three queries, which differ only in the team mentioned in the WHERE clause.

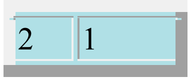
```
1)   SELECT COUNT(*), COUNT(Runs)
     FROM Scores
     WHERE Team = 'Tigers'

2)   SELECT COUNT(*), COUNT(Runs)
     FROM Scores
     WHERE Team = 'Carp'

3)   SELECT COUNT(*), COUNT(Runs)
     FROM Scores
     WHERE Team = 'Swallows'
```

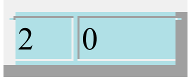
Question Explanation: Each of the queries selects two rows of the table; for example, query (1) selects rows 2 and 8. Thus, COUNT(*) produces 2 for each query. COUNT(Runs) produces the number of rows that do not have NULL in the Runs column.

For query (1), only row 2 does not have NULL, so the tuple produced is:



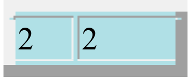
2	1

For query (2), both matching tuples, 3 and 9, have NULL in the Runs column. When there is nothing over which to take an aggregate, the result is NULL, with the exception of the COUNT operator, where the count is then 0. Thus, query (2) produces:



2	0

For query (3), both matching rows, 4 and 10, are non-NULL in the Runs column, so COUNT(Runs) is 2. The tuple produced is:



2	2

Question Text: For what values of *x*, *y*, and *z*, including NULL, does the boolean expression

```
x <= 3 AND NOT(y >= 1 OR z = 5)
```

have the truth value UNKNOWN?

Question Explanation: The AND of expressions has the value UNKNOWN if and only if:

1. Neither side is FALSE.
2. At least one side is UNKNOWN.

The only way for the left side to be UNKNOWN is if *x* = NULL. In that case, the right side can be either TRUE or UNKNOWN. Since the right side is the NOT of an expresion, that expression, `(y >= 1 OR z = 5)` can be either FALSE or UNKNOWN. An OR of expressions is TRUE if and only if either side is true. Thus, if *x* = NULL, then we must not have $y \geq 1$, and we must not have $z=5$. All other values for *y* and *z*, including NULL, are OK.

The only other case to consider is if $x \leq 3$ is TRUE. Then `y >= 1 OR z = 5` must be UNKNOWN. That is, at least one of y and z is NULL and neither $y \geq 1$ nor $z = 5$ is TRUE. In summary, the possible values of x, y, and z are:

1. x = NULL, $y < 1$, $z \neq 5$.
2. x = NULL, y = NULL, $z \neq 5$.
3. x = NULL, $y < 1$, z = NULL.
4. x = NULL, y = NULL, z = NULL.
5. $x \leq 3$, $y < 1$, z = NULL.
6. $x \leq 3$, y = NULL, $z \neq 5$.
7. $x \leq 3$, y = NULL, z = NULL.

Question Text: For what values of x, y, and z, including NULL, does the boolean expression

```
x <= 3 OR NOT(y >= 1 AND z = 5)
```

have the truth value FALSE?

Question Explanation: The OR of expressions has the value FALSE if and only if both sides have the value FALSE. Thus, we know that we must have x > 3; NULL is not a possible value for x, because that would make the entire expression at least UNKNOWN.
For NOT(y >= 1 AND z = 5) to be FALSE, (y >= 1 AND z = 5) must be TRUE. That is, y ≥ 1 and z = 5 must both be TRUE.