# Constraints and Triggers Quiz

Bookmark this page

Quiz due May 4, 2022 22:28 IST

Each multiple-choice quiz problem is based on a "root question," from which the system generates different correct and incorrect choices each time you take the quiz. Thus, you can test yourself on the same material multiple times. We strongly urge you to continue testing on each topic until you complete the quiz with a perfect score at least once. Simply click the "Reset" button at the bottom of the page for a new variant of the quiz.

After submitting your selections, the system will score your quiz, and for incorrect answers will provide an "explanation" (sometimes for correct ones too). These explanations should help you get the right answer the next time around. To prevent rapid-fire guessing, the system enforces a minimum of 10 minutes between each submission of solutions.

---

## Q1

1/1 point (graded)

[Q1] Consider the following SQL table declaration:

```
CREATE TABLE R (a INT, b INT, c INT, CHECK( [fill-in] ));
```

Currently R contains the tuples (1,4,14), (2,3,15), and (3,3,16). Which of the following tuple-based CHECK constraints will cause the following insertion to be rejected?

```
INSERT INTO R VALUES (4,4,9);
```

Note: When a tuple-based check is invoked for an insert and includes a subquery over the same table, the subquery is evaluated on the table *including* the inserted tuple.

- ○ b < (SELECT MIN(c) FROM R)
- ◉ c >= (SELECT SUM(b) FROM R)
- ○ c > ALL (SELECT a + b FROM R)
- ○ a <= ALL (SELECT c - b FROM R)

✔

**Problem Explanation**

The insertion is rejected when the CHECK condition is false. Note that an attribute mentioned outside of a subquery refers to the inserted tuple. Substititue the inserted tuple's values for those variables, and evaluate the expression over the table; make sure to include the inserted tuple in any subqueries. Thus, for example, SUM(b) currently has the value 14 (including the inserted tuple), COUNT(c) has the value 4, and the result of "SELECT b+c FROM R" is {18,18,19,13}.

Submit

Hide Notes

Submit

Answers are displayed within the problem

## Q2

1/1 point (graded)

[Q2] Consider the following trigger over a table R(a,b), specified using the trigger language of the SQL standard:

```
CREATE TRIGGER Rins
AFTER INSERT ON R
REFERENCING NEW ROW AS new
FOR EACH ROW
INSERT INTO R(a,b)
  (SELECT DISTINCT R.a, new.b
   FROM R
   WHERE R.b = new.a)
  EXCEPT
  (SELECT DISTINCT a,b FROM R)
```

Suppose table R is empty initially. We issue three commands to insert tuples into R: first we insert (1,2), then we insert (2,3), then we insert (3,4). After some of these inserts, trigger **Rins** may insert further tuples into R, which may activate the trigger recursively. After all the inserts are done, which of these tuples is NOT in table R?

- (4,3)
- (1,4)
- (2,4)
- (1,3)

✔

**Problem Explanation**

The trigger looks for existing tuples with a **b** value equal to the **a** value of the newly inserted tuple. It inserts new tuples with **a** values from the matching existing tuples and the **b** value from the new tuple. The EXCEPT part ensures that tuples are not inserted when the same tuple is already present.
Inserting (1,2) does not trigger the insertion of any other tuples. Inserting (2,3) triggers the insertion of (1,3), since the new **a** value of 2 matches the **b** value of the existing tuple (1,2). Inserting (1,3) does not (recursively) trigger the insertion of any other tuples. Inserting (3,4) triggers the insertion of (1,4) and (2,4), since the new **a** value of 3 matches the **b** value of the existing tuples (1,3) and (2,3). Inserting (1,4) does not (recursively) trigger the insertion of any other tuples. Inserting (2,4) (recursively) triggers the insertion of (1,4), since the new **a** value of 2 matches the **b** value of the existing tuple (1,2), but (1,4) is already present so a new copy is not inserted.

Submit

Hide Notes

Answers are displayed within the problem

## Q3

1/1 point (graded)
[Q3] Consider the following SQL table declaration:

```
   CREATE TABLE Emps (id INT, ssNo INT, name CHAR(20),
managerID INT);
```

We would like to extend the table declaration to enforce that each of **id** and **ssNo** is a key (by itself), and each value of **managerID** must be one of the values that appears in the **id** attribute of the same table. Which of the following is *not* a legal addition of SQL standard key and/or foreign-key constraints? Note: The addition does not have to achieve all of the stated goals; it only must result in legal SQL.

- Add "UNIQUE" after each of the first two INT's.
- Add "PRIMARY KEY" after the first INT, and add "UNIQUE" after the second INT.
- Add "UNIQUE" after the first INT, and add "PRIMARY KEY" after the second INT.
- ✔ Add ", FOREIGN KEY (managerID) REFERENCES Emps(id)" before the closing parenthesis.

**Answer-Selection Feedback**
A foreign key must reference a declared key. While **id** is intended to be a key, we have not declared it as such in this modification.

**Problem Explanation**
The correct answers (i.e., incorrect SQL) violate one of two rules:

1. You cannot have two primary keys (although you can have many "uniques").
2. A foreign key can only reference an attribute that is a key (either primary key or unique).

Submit

Answers are displayed within the problem

## Q4

1/1 point (graded)
[Q4] Here are SQL declarations for two tables S and T:

```
   CREATE TABLE S(c INT PRIMARY KEY, d INT);
   CREATE TABLE T(a INT PRIMARY KEY, b INT REFERENCES S(c));
```

Suppose S(c,d) contains four tuples: (2,10), (3,11), (4,12), (5,13).
Suppose T(a,b) contains four tuples: (0,4), (1,5), (2,4), (3,5). As a

Hide Notes

Suppose T(a,b) contains four tuples: (0,4), (1,5), (2,4), (3,5). As a result of the constraints in the table declarations, certain insertions, deletions, and/or updates on S and T are disallowed. Which of the following modifications will *not* violate any constraint?

- ○ Inserting (1,2) into T
- ○ Deleting (4,12) from S
- ○ Inserting (0,3) into T
- ◉ Deleting (0,4) from T ✔

**Problem Explanation**
The question choices explore insertions into both tables and deletions from S. Since attribute **c** is a key for S, we cannot insert into S any tuple with first component 2, 3, 4, or 5. Since attribute **a** is a key for T, we cannot insert into T any tuple with first component 0, 1, 2, or 3. Since attribute **T.b** is a foreign key referencing **S.c**: (1) An insertion into T must have second component 2, 3, 4, or 5; (2) We cannot delete from S any tuple whose first component is a second component of T, that is, 4 or 5.

Submit

ⓘ Answers are displayed within the problem

## Q5

1/1 point (graded)
[Q5] Here are SQL declarations for two tables S and T:

```
   CREATE TABLE S(c INT PRIMARY KEY, d INT);
   CREATE TABLE T(a INT PRIMARY KEY, b INT, CHECK(b IN
(SELECT c FROM S)));
```

Suppose S(c,d) contains the four tuples: (2,10), (3,11), (4,12), (5,13). Suppose T(a,b) contains the four tuples: (0,4), (1,5), (2,4), (3,5). As a result of the constraints in the table declarations, certain insertions, deletions, and/or updates on S and T are disallowed. Which of the following modifications will *not* violate any constraint?

- ○ Inserting (1,4) into T
- ◉ Updating (2,4) in T to be (2,2) ✔
- ○ Updating (3,5) in T to be (3,1)
- ○ Inserting (5,0) into T

**Problem Explanation**
The question choices explore insertions into both tables and updates to T. Since attribute **c** is a key for S, we cannot insert into

S any tuple with first component 2, 3, 4, or 5. Since attribute **a** is a key for T, we cannot insert into T any tuple with first component 0, 1, 2, or 3. The CHECK constraint says: (1) An insertion into T must have second component 2, 3, 4, or 5; (2) An update to T must keep the value of the second component as one of 2, 3, 4, or 5.

Submit

Answers are displayed within the problem

## Q6

1/1 point (graded)

[Q6] Consider the following trigger over a table R(a,b), specified using the trigger language of the SQL standard:

```
CREATE TRIGGER Rins
AFTER INSERT ON R
REFERENCING NEW ROW AS new
FOR EACH ROW
WHEN (new.a * new.b > 10)
INSERT INTO R VALUES (new.a - 1, new.b + 1);
```

When we insert a tuple into R, the trigger may cause another tuple to be inserted, which may cause yet another tuple to be inserted, and so on, until finally a tuple is inserted that does not cause the trigger to fire. Suppose we begin with table R empty. Consider the following possible tuples inserted into R. After trigger execution completes, which of the insertions results in R containing exactly 3 tuples?

- (3,5)
- (5,4)
- (50,0)
- (2,6)

**Problem Explanation**

Each time the trigger is activated by an insertion (x,y), if x*y is greater than 10 then the trigger inserts a new tuple (x-1,y+1). Thus, if we insert (x,y) first, then in order to insert exactly three tuples, we need for x*y and (x-1)*(y+1) to be greater than 10, but for (x-2)*(y+2) to be 10 or less.

Submit

Answers are displayed within the problem

## Q7

1/1 point (graded)

Hide Notes

1/1 point (graded)

[Q7] Here are SQL declarations for three tables R, S, and T:

```
   CREATE TABLE R(e INT PRIMARY KEY, f INT);
   CREATE TABLE S(c INT PRIMARY KEY, d INT REFERENCES R(e) ON
DELETE CASCADE);
   CREATE TABLE T(a INT PRIMARY KEY, b INT REFERENCES S(c) ON
DELETE CASCADE);
```

Suppose R(e,f) contains tuples (1,0), (2,4), (3,5), (4,3), and (5,7). Suppose S(c,d) contains tuples (1,5), (2,2), (3,3), (4,5), and (5,4). Suppose T(a,b) contains tuples (0,2), (1,2), (2,3), (3,4), and (4,4). As a result of the referential integrity actions in the table declarations, certain deletions may cause additional deletions to be performed automatically. Which of the following deletions, after all integrity actions, leaves table T empty?

- delete from R where e+f<=8
- delete from R where f<6
- delete from R where e*f>=10
- delete from R where e=f-2 ✔

**Problem Explanation**

T contains tuples (0,2), (1,2), (2,3), (3,4), and (4,4). The removal of any of the following elements will result in the deletion of (0,2) from T: (0,2) from T, (2,2) from S, or (2,4) from R. The removal of any of the following elements will result in the deletion of (1,2) from T: (1,2) from T, (2,2) from S, or (2,4) from R. The removal of any of the following elements will result in the deletion of (2,3) from T: (2,3) from T, (3,3) from S, or (3,5) from R. The removal of any of the following elements will result in the deletion of (3,4) from T: (3,4) from T, (4,5) from S, or (5,7) from R. The removal of any of the following elements will result in the deletion of (4,4) from T: (4,4) from T, (4,5) from S, or (5,7) from R. For T to be empty, any deletion must remove at least one element from each set.

Submit

‹ Previous  Next ›