

CSC343 Worksheet 5 Solution

June 19, 2020

```
1. a) CREATE TABLE MovieExec (  
2     name CHAR(30),  
3     address VARCHAR(255),  
4     cert# INT PRIMARY KEY,  
5     FOREIGN KEY (cert#) REFERENCES Movies(producerC#)  
6 );  
7
```

Example:

- Foreign-key
 - **Syntax 1:** FOREIGN KEY (< attributes >) REFERENCES < table >(< attributes >)
 - **Syntax 2:** REFERENCES < table >(< attributes >)
 - Binds an attribute of one relation to an attribute in another table
 - Added when creating table

Example:

```
1 // Example 1  
2 CREATE TABLE Studio (  
3     name CHAR(30) PRIMARY KEY,  
4     address VARCHAR(255),  
5     presC# INT REFERENCES MovieExec(cert#)  
6 );  
7  
8 // Example 2  
9 CREATE TABLE Studio (  
10    name CHAR(30) PRIMARY KEY,  
11    address VARCHAR(255),  
12    presC# INT,  
13    FOREIGN KEY (presC#) REFERENCES MovieExec(cert#)  
14 );  
15
```

b)

```

1  CREATE TABLE Movies (
2      title CHAR(30) PRIMARY KEY,
3      year INT PRIMARY KEY,
4      length INT,
5      genre VARCHAR(255),
6      studioName VARCHAR(255),
7      producerC# PRIMARY KEY
8  );
9

```

c) No change required. Violation occurs by the default policy.

```

1  CREATE TABLE MovieExec (
2      name CHAR(30),
3      address VARCHAR(255),
4      cert# INT PRIMARY KEY,
5      FOREIGN KEY (cert#) REFERENCES Movies(producerC#)
6  );
7

```

Correct Solution:

```

1  CREATE TABLE MovieExec (
2      name CHAR(30),
3      address VARCHAR(255),
4      cert# INT PRIMARY KEY,
5      FOREIGN KEY (cert#) REFERENCES Movies(producerC#)
6          ON UPDATE CASCADE // Correction
7          ON DELETE CASCADE // Correction
8  );
9

```

Notes:

- Maintaining Referential Integrity
 - Three different types of policies exist on Foreign Key
 1. *The Default Policy: Reject Violating Modifications.*
 - * Is default policy
 - * Rejects any modification violating referential integrity constant
 2. *The Cascade Policy*
 - * Changes to the referenced attributes are mimicked at foreign key.
 - * e.g. delete a tuple in **MovieExec**, deletes related referencing tuple(s) from **Studio**
 3. *The Set-Null Policy*
 - * When a modification to the referenced relation affects a foreign-key value, the latter is changed to NULL.

* This applies to both UPDATE and DELETE

Example:

```
1 CREATE TABLE Movies (  
2     title CHAR(30) PRIMARY KEY,  
3     year INT PRIMARY KEY,  
4     length INT,  
5     genre VARCHAR(255),  
6     studioName VARCHAR(255),  
7     producerC# REFERENCES MovieExec(cert#)  
8         ON DELETE SET NULL  
9         ON UPDATE CASCADE  
10 );  
11
```

d)

```
1 CREATE TABLE Movies (  
2     title CHAR(30) PRIMARY KEY,  
3     year INT PRIMARY KEY,  
4     length INT,  
5     genre VARCHAR(255),  
6     studioName VARCHAR(255),  
7     producerC# VARCHAR(255)  
8     FOREIGN KEY (title) REFERENCES StarsIn(movieTitle)  
9 );  
10
```

e)

```
1 CREATE TABLE Movies (  
2     title CHAR(30) PRIMARY KEY,  
3     year INT PRIMARY KEY,  
4     length INT,  
5     genre VARCHAR(255),  
6     studioName VARCHAR(255),  
7     producerC# VARCHAR(255)  
8     FOREIGN KEY (title) REFERENCES StarsIn(movieTitle)  
9 );  
10
```