**Constraints Movie-Ratings Exercises**

关文聪 2016060601008

（已更新到个人github，详见：https://github.com/Eternity-Myth/Database-CourseHomework/tree/master/3-Constraints%20Movie-Ratings）

You will enhance the movie-ratings database that was also used for the SQL Movie-Ratings Query Exercises. In this set of exercises you will declare integrity constraints on the data, and you will verify that they are being enforced by the underlying database management system. You will experiment with several types of constraints: *key constraints*, *non-null constraints*, *attribute-based*and*tuple-based check constraints*, and *referential integrity*. A SQL file to set up the original schema and data for the movie-ratings database is downloadable [here](https://prod-c2g.s3.amazonaws.com/db/Winter2013/files/rating.sql). You will be using the same data, but modifying the schema to add constraints. The original schema and data can be loaded as specified in the file into SQLite, MySQL, or PostgreSQL. However, currently MySQL does not enforce constraints (even though it accepts some of them syntactically). For these exercises, currently you must use SQLite or PostgreSQL. See our [quick guide](https://class.stanford.edu/c4x/Engineering/db/asset/sql-guide.html) for installing and using all three systems.

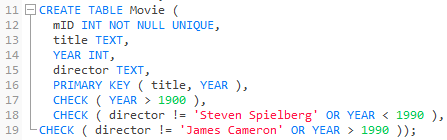
**说明：**

**注意到MySQL中不支持Check语句（或者说Check语句在数据库实际运行过程中不起约束作用），因此，本次实验将全部采用PostgreSQL数据库完成，其对应的版本为：postgresql-11.0-1-windows-x64，安装数据库与环境配置的步骤此处省略。**

**Schema:**  
Movie ( mID, title, year, director )  
English: There is a movie with ID number *mID*, a *title*, a release *year*, and a *director*.  
  
Reviewer ( rID, name )  
English: The reviewer with ID number *rID* has a certain *name*.  
  
Rating ( rID, mID, stars, ratingDate )  
English: The reviewer *rID* gave the movie *mID* a number of *stars* rating (1-5) on a certain *ratingDate*.  
  
Unlike most of our other exercises, which are a set of queries to be written individually, this exercise set involves bigger chunks of work followed by a series of tests. If the constraints are implemented correctly, the tests will generate or not generate errors as specified. To verify that the referential integrity policies are implemented correctly, there is a check of the final database state.  
**Task 1: Constraint Declarations**Modify the three CREATE TABLE statements in the [movie-rating database](https://prod-c2g.s3.amazonaws.com/db/Winter2013/files/rating.sql) to add the following ten constraints. (Note: You may want to examine the date format in the data file so you can specify date-related constraints as string comparisons.)  
  
**Key Constraints**  
  
1.  mID is a key for Movie  
2.  (title,year) is a key for Movie  
3.  rID is a key for Reviewer  
4.  (rID,mID,ratingDate) is a key for Rating but with null values allowed  
  
**Non-Null Constraints**  
  
5.  Reviewer.name may not be NULL  
6.  Rating.stars may not be NULL  
  
**Attribute-Based Check Constraints**  
  
7.  Movie.year must be after 1900  
8.  Rating.stars must be in {1,2,3,4,5}  
9.  Rating.ratingDate must be after 2000  
  
**Tuple-Based Check Constraints**  
  
10.  "Steven Spielberg" movies must be before 1990 and "James Cameron" movies must be after 1990

根据以上条件，修改Create Table语句，为以上属性添加约束，修改后的Create Table语句如下所示：

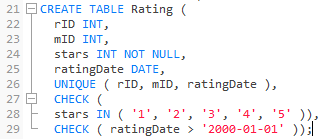
对Movie表：



对Revier表：

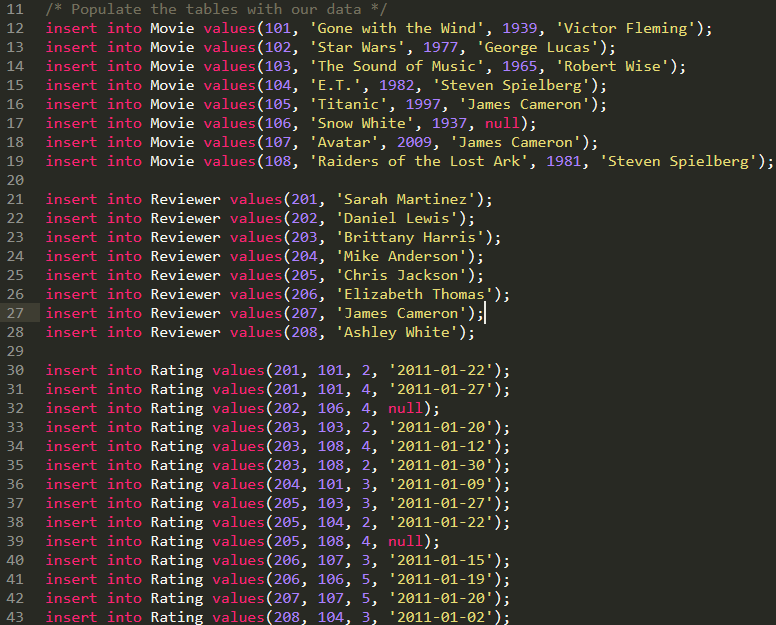


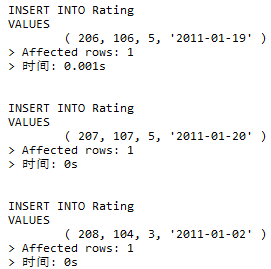
对Rating表：



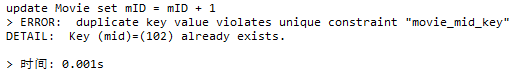
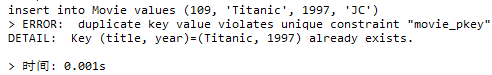
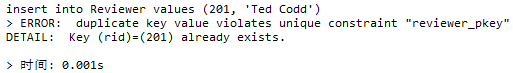
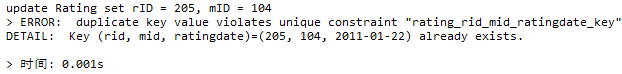
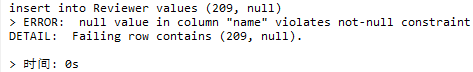
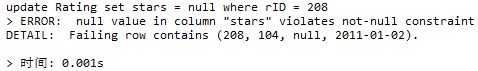
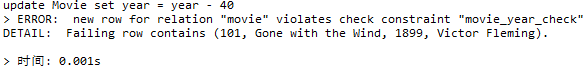
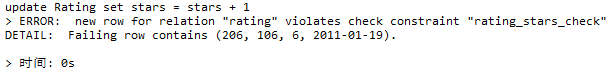
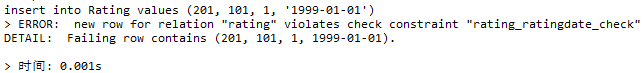
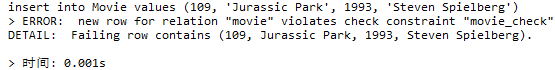
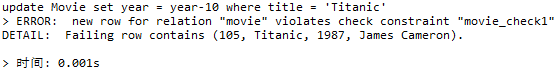
**Task 2: Load the Database**  
  
After creating the three tables using your modified CREATE TABLE statements, you should be able to load the original data (i.e., execute all of the INSERT statements in the data file) without any errors.

检验：将原始数据导入，即执行如下图所示的SQL语句插入数据，所有SQL语句应该顺利执行完成无任何错误。





从执行结果看，所有SQL语句都顺利执行完成，无任何错误，可以继续进行下一步的实验。

**Task 3: Constraint Enforcement**  
  
*Each of the following commands should generate an error.*  
  
11.  update Movie set mID = mID + 1;  
  
12.  insert into Movie values (109, 'Titanic', 1997, 'JC');  
  
13.  insert into Reviewer values (201, 'Ted Codd');  
  
14.  update Rating set rID = 205, mID = 104;  
  
15.  insert into Reviewer values (209, null);  
  
16.  update Rating set stars = null where rID = 208;  
  
17.  update Movie set year = year - 40;  
  
18.  update Rating set stars = stars + 1;  
  
19.  insert into Rating values (201, 101, 1, '1999-01-01');  
  
20.  insert into Movie values (109, 'Jurassic Park', 1993, 'Steven Spielberg');  
  
21.  update Movie set year = year-10 where title = 'Titanic';  


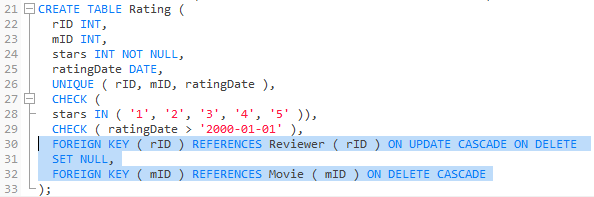
由得到的这些结果可以看出，上述所有的SQL语句在运行过程中都会产生错误，无法正常运行，这验证了之前创建表时添加的约束的正确性，是符合实验预期的正确结果，因此，可以继续进行下一步的实验。

*None of the following commands should generate errors.*  
  
22.  insert into Movie values (109, 'Titanic', 2001, null);  
  
23.  update Rating set mID = 109;  
  
24.  update Movie set year = 1901 where director <> 'James Cameron';  
  
25.  update Rating set stars = stars - 1;  

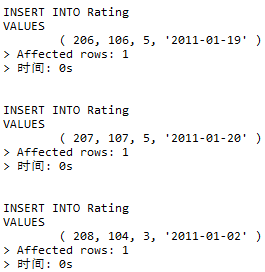

由得到的这些结果可以看出，上述所有的SQL语句在运行过程中均可以正常运行完成，没有出现任何错误。这是符合实验预期的正确结果，因此，可以继续进行下一步的实验。

**Task 4: Referential Integrity Declarations**  
  
Further modify one or more of your CREATE TABLE statements to include the following referential integrity constraints and policies.  
  
26.  Referential integrity from Rating.rID to Reviewer.rID  
          Reviewers updated: cascade  
          Reviewers deleted: set null  
          All others: error  
  
26.  Referential integrity from Rating.mID to Movie.mID  
          Movies deleted: cascade  
          All others: error

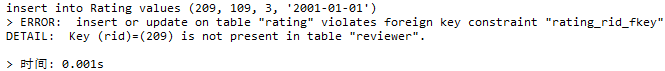
由题意，需要修改Rating表中的rID与mID属性，将这两个属性设置为外键（Foreign Key）约束，被参照键分别为Reviewer表中的rID属性与Movie表中的mID属性，并设置相应的处理策略。修改后的Create Table语句如下所示：

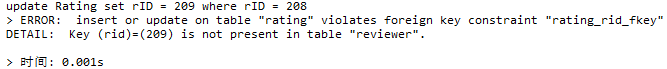
  
  
**Task 5: Reload the Database**  
  
Recreate the three tables using your modified CREATE TABLE statements. You should be able to load the original data (i.e., execute all of the INSERT statements in the data file) without any errors.

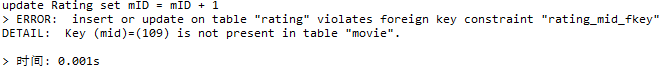
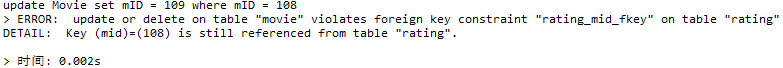
检验：将原始数据导入（与上文的原始数据相同），所有SQL语句应该顺利执行完成无任何错误。



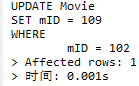
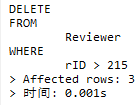
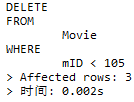
从执行结果看，所有SQL语句都顺利执行完成，无任何错误，可以继续进行下一步的实验。

**Task 6: Referential Integrity Enforcement**  
  
*Each of the following commands should generate an error.*  
 **Important Note: If using SQLite, make sure to turn on referential integrity checking with the command "pragma foreign\_keys = on;"**  
  
27.  insert into Rating values (209, 109, 3, '2001-01-01');  


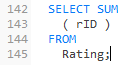
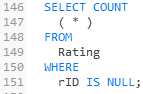
28.  update Rating set rID = 209 where rID = 208;  


29.  update Rating set mID = mID + 1;  
  
30.  update Movie set mID = 109 where mID = 108;  


由得到的这些结果可以看出，上述所有的SQL语句在运行过程中都会产生错误，无法正常运行，这验证了之前创建表时添加的约束的正确性，是符合实验预期的正确结果，因此，可以继续进行下一步的实验。

*None of the following commands should generate errors, but they will make additional database modifications according to the referential-integrity policies.*31.  update Movie set mID = 109 where mID = 102;  
  
32.  update Reviewer set rID = rID + 10;  
  
33.  delete from Reviewer where rID > 215;  
  
34.  delete from Movie where mID < 105;

由得到的这些结果可以看出，上述所有的SQL语句在运行过程中均可以正常运行完成，没有出现任何错误。这是符合实验预期的正确结果，因此，可以继续进行下一步的实验。

**Final Check**  
  
35.  Check the resulting database by writing SQL queries to compute:  
          (a) The sum of non-null rIDs in the Rating table -- should be 853    
          (b) The number of tuples in Rating with null rIDs -- should be 3  
 

如图所示，执行完上述的语句后，执行上图所示的两个查询语句，得到了与实验预期结果一样的结果，这验证了以上实验操作的正确性。这说明之前设置的约束语句是正确的。