495 Project2 finished by XinTONG

Q1:

Enforce the constraint that categories of a movie must be either "Romantic", "Comedy", "Drama", or "Action". Suppose, the default value for the Category field is "Action". If a non-allowed value is inserted/updated, the category for that tuple must be changed to the default value.

```
DROP TRIGGER if exists m_category1;
delimiter //
CREATE TRIGGER m_category1
BEFORE UPDATE ON made_money
FOR EACH ROW
BEGIN
IF new.category=NULL OR
(new.category<> 'Romantic' AND new.category<> 'Comedy' AND
new.category<>'Drama' AND new.category<>'Action')
THEN
SET new.category='Action';
END IF;
END;
//
delimiter ;

DROP TRIGGER if exists m_category2;
delimiter //
CREATE TRIGGER m_category2
BEFORE INSERT ON made_money
FOR EACH ROW
BEGIN
IF new.category=NULL OR
(new.category<> 'Romantic' AND new.category<> 'Comedy' AND
new.category<>'Drama' AND new.category<>'Action')
THEN
SET new.category='Action';
END IF;
END;
//
delimiter ;
```

Q2:

Enforce the following condition: A star can only be a part of a "Comedy" movie, only if he/she has performed in at least one "Romantic", "Comedy", or "Drama" movie previously. Upon insertion of a tuple violating this (e.g., a Comedy movie associated with a star who has previously done only "Action" movies), the category of the movie

must be updated to "Drama".

```
DROP TRIGGER if exists star_movie;
delimiter //
CREATE TRIGGER star_movie AFTER INSERT ON appeared_in
FOR EACH ROW BEGIN
IF (NOT EXISTS (SELECT *
FROM made_money, appeared_in WHERE made_money.movie=appeared_in.movie
AND new.star=appeared_in.star
AND new.movie<>appeared_in.movie
AND (made_money.category ='Romantic'
OR made_money.category ='Comedy'
OR made_money.category ='Drama')
AND made_money.day_opened<
(SELECT made_money.day_opened from made_money WHERE
made_money.movie=new.movie)))
THEN UPDATE made_money SET made_money.category='DRAMA' WHERE
made_money.category='comedy' AND made_money.movie=new.movie ;
END IF;
END;
// delimiter ;
```

Q3:
Enforce the constraint: A star cannot be married to multiple stars simultaneously.

```
DROP TRIGGER if exists star_marriage;
delimiter //
CREATE TRIGGER star_marriage AFTER INSERT ON married
FOR EACH ROW BEGIN
IF EXISTS(

select * from in_couple A where A. COUPLE_NUM not in
(select D. COUPLE_NUM from divorce D inner join married E on D.COUPLE_NUM =
E.COUPLE_NUM)
and new.COUPLE_NUM=A.COUPLE_NUM)
THEN
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = "ERROR: Check constraint on married.";
END IF;
END;
// delimiter ;
```

Q4:
Enforce that, a movie must make at least $1,000 in the box office, and cannot make

more than 3 billion ($3,000,000,000) in the box office. Also, if a movie category is "Action", then it should make at least $10,000, and if category is "Comedy", it cannot make more than $1,000,000,000.

```
DROP TRIGGER if exists movie_money;
delimiter //
CREATE TRIGGER movie_money BEFORE INSERT ON made_money
FOR EACH ROW
BEGIN
IF EXISTS(
select * from made_money A
where new.HOW_MUCH<1000
or new.how_much>3000000000
or (new.how_much<10000 and new.Category='Action')
or (new.how_much>1000000000 and new.Category='Comedy'))
THEN
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = " ERROR: Check constraint on made money.";
END IF;
END;
// delimiter ;
```

Q5:

Using a trigger, ensure that the divorce date of a couple is at least the same or after their marriage date. If this is violated, set the divorce date to be the same as the marriage date.

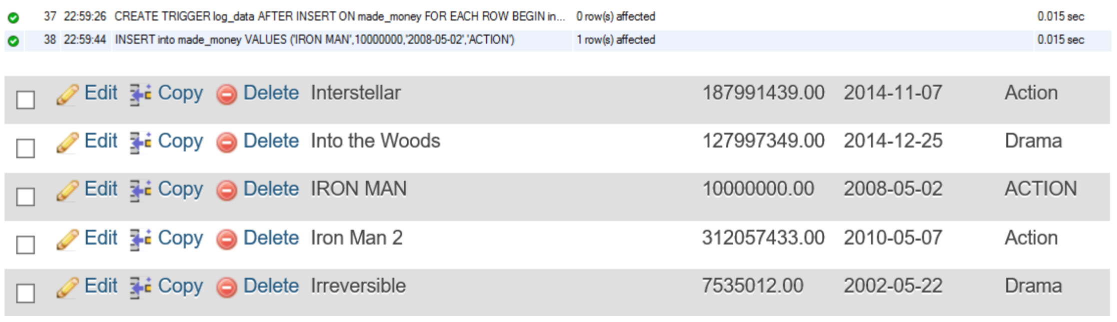
```
DROP TRIGGER if exists Divorce_condition;
delimiter //
CREATE TRIGGER Divorce_condition BEFORE INSERT ON divorced
FOR EACH ROW BEGIN
IF EXISTS(
select * from married
where married.couple_num=new.couple_num
and new.day<married.day)
THEN
set new.day=(select married.day from married where married.COUPLE_NUM = new.COUPLE_NUM);
END IF;
END;
// delimiter ;
```

Q6:
We want to keep a log file containing data (movie & category) from rows that have been inserted into “MADE_MONEY” table into the given “LOG_DATA” table. Use a trigger to accomplish this goal.

```
DROP TRIGGER if exists log_data;
delimiter //
CREATE TRIGGER log_data AFTER INSERT ON made_money
FOR EACH ROW
BEGIN
insert into log_data
values (new.MOVIE, new.Category);
END;
// delimiter ;
```

a) Insert a new movie, with values (“IRON MAN ” , 1000000, 2008-05-02, “ACTON ”) in MADE_MONEY table.

```
INSERT into made_money
VALUES ('IRON MAN',10000000,'2008-05-02','ACTION')
```

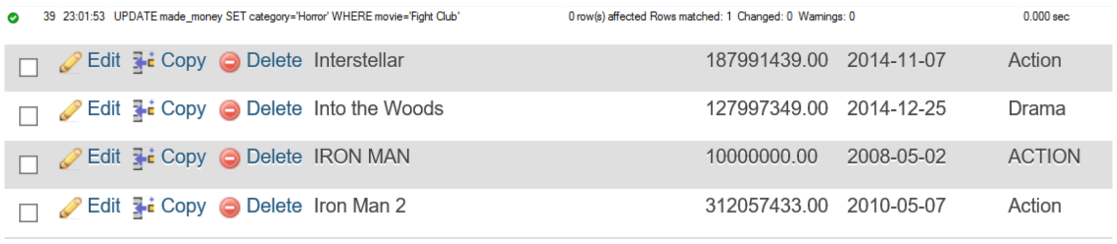


```
37  22:59:26  CREATE TRIGGER log_data AFTER INSERT ON made_money FOR EACH ROW BEGIN in...  0 row(s) affected  0.015 sec
38  22:59:44  INSERT into made_money VALUES ('IRON MAN',10000000,'2008-05-02','ACTION')     1 row(s) affected  0.015 sec
```

Edit	Copy	Delete	Interstellar	187991439.00	2014-11-07	Action
Edit	Copy	Delete	Into the Woods	127997349.00	2014-12-25	Drama
Edit	Copy	Delete	IRON MAN	10000000.00	2008-05-02	ACTION
Edit	Copy	Delete	Iron Man 2	312057433.00	2010-05-07	Action
Edit	Copy	Delete	Irreversible	7535012.00	2002-05-22	Drama

b) Update the CATEGORY of the movie “Fight Club ” to “Horror in MADE_MONEY table.

```
UPDATE made_money
SET category='Horror' WHERE movie='Fight Club'
```

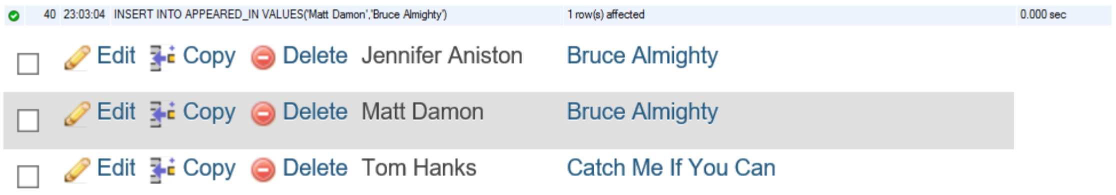


```
39  23:01:53  UPDATE made_money SET category='Horror' WHERE movie='Fight Club'   0 row(s) affected Rows matched: 1  Changed: 0  Warnings: 0   0.000 sec
```

Edit	Copy	Delete	Interstellar	187991439.00	2014-11-07	Action
Edit	Copy	Delete	Into the Woods	127997349.00	2014-12-25	Drama
Edit	Copy	Delete	IRON MAN	10000000.00	2008-05-02	ACTION
Edit	Copy	Delete	Iron Man 2	312057433.00	2010-05-07	Action

c) Insert a new tuple in APPEARED_IN table, with values (“Matt Damon ” , “Bruce Almighty ”).

INSERT INTO APPEARED_IN
VALUES('Matt Damon','Bruce Almighty')

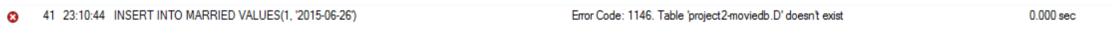


```
40  23:03:04  INSERT INTO APPEARED_IN VALUES('Matt Damon','Bruce Almighty')      1 row(s) affected      0.000 sec
```

☐	Edit	Copy	Delete	Jennifer Aniston	Bruce Almighty
☐	Edit	Copy	Delete	Matt Damon	Bruce Almighty
☐	Edit	Copy	Delete	Tom Hanks	Catch Me If You Can

d) Insert a new tuple in MARRIED, with values (1, 2015-06-26).

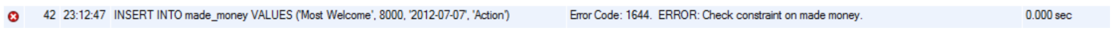
INSERT INTO MARRIED
VALUES(1, '2015-06-26')



```
41  23:10:44  INSERT INTO MARRIED VALUES(1, '2015-06-26')      Error Code: 1146. Table 'project2-moviedb.D' doesn't exist      0.000 sec
```

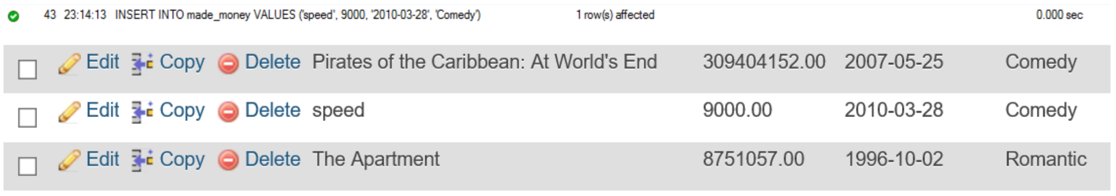
e) Insert two new tuples in MADE_MONEY, having values (“Most Welcome” , 8000, 2012-07-07, “Action”) and (“Speed” , 9000, 2010-03-28, “Comedy”).

INSERT INTO made_money
VALUES ('Most Welcome', 8000, '2012-07-07', 'Action')



```
42  23:12:47  INSERT INTO made_money VALUES ('Most Welcome', 8000, '2012-07-07', 'Action')      Error Code: 1644.  ERROR: Check constraint on made money.      0.000 sec
```

INSERT INTO made_money
VALUES ('speed', 9000, '2010-03-28', 'Comedy')

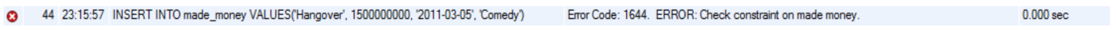


```
43  23:14:13  INSERT INTO made_money VALUES ('speed', 9000, '2010-03-28', 'Comedy')      1 row(s) affected      0.000 sec
```

☐	Edit	Copy	Delete	Pirates of the Caribbean: At World's End	309404152.00	2007-05-25	Comedy
☐	Edit	Copy	Delete	speed	9000.00	2010-03-28	Comedy
☐	Edit	Copy	Delete	The Apartment	8751057.00	1996-10-02	Romantic

f) Insert a new tuple in MADE_MONEY, having values (“Hangover” , 1500000000, 2011-03-05, “Comedy”).

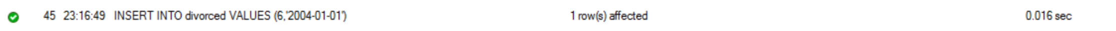
INSERT INTO made_money
VALUES('Hangover', 1500000000, '2011-03-05', 'Comedy')



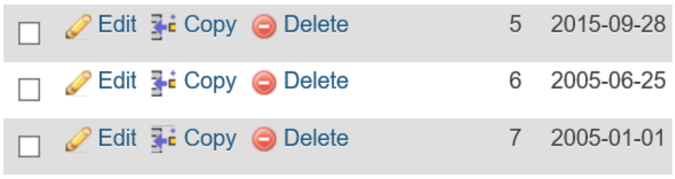
```
44  23:15:57  INSERT INTO made_money VALUES('Hangover', 1500000000, '2011-03-05', 'Comedy')      Error Code: 1644.  ERROR: Check constraint on made money.      0.000 sec
```

g) Insert a new tuple in DIVORCED, with values (6, 2004-01-01)

INSERT INTO divorced
VALUES (6,'2004-01-01')



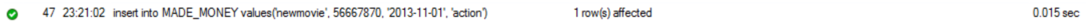
```
45  23:16:49  INSERT INTO divorced VALUES (6,'2004-01-01')      1 row(s) affected      0.016 sec
```



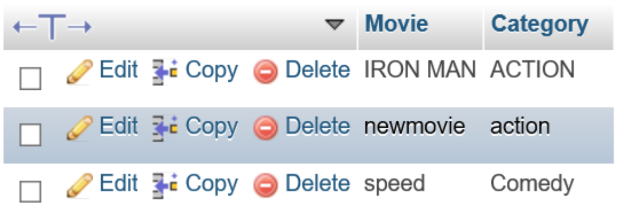
☐	Edit	Copy	Delete	5	2015-09-28
☐	Edit	Copy	Delete	6	2005-06-25
☐	Edit	Copy	Delete	7	2005-01-01

Test Q6:

insert into MADE_MONEY
values('newmovie', 56667870, '2013-11-01', 'action')



```
47  23:21:02  insert into MADE_MONEY values('newmovie', 56667870, '2013-11-01', 'action')    1 row(s) affected    0.015 sec
```



←T→				Movie	Category
☐	Edit	Copy	Delete	IRON MAN	ACTION
☐	Edit	Copy	Delete	newmovie	action
☐	Edit	Copy	Delete	speed	Comedy