Project2_Answer

By Hanzhi Yang

1. Syntax fro the trigger

1.1. Question 1

```
DROP TRIGGER IF EXISTS `project2-moviedb`.`movie_category_insert`;

DELIMITER $$
USE `project2-moviedb`$$
CREATE DEFINER=`root`@`localhost` TRIGGER `movie_category_insert` BEFORE
INSERT ON `made_money` FOR EACH ROW begin
        if 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 `project2-moviedb`.`movie_category_update`;

DELIMITER $$
USE `project2-moviedb`$$
CREATE DEFINER=`root`@`localhost` TRIGGER `movie_category_update` BEFORE
UPDATE ON `made_money` FOR EACH ROW begin
        if new.category != 'Romantic' and new.category != 'Comedy' and
new.category != 'Drama' and new.category !='Action'
    then set new.category = 'Action';
    end if;
end$$
DELIMITER ;
```

1.2. Question 2

```
DROP TRIGGER IF EXISTS `project2-moviedb`.`appeared_in_BEFORE_INSERT`;

DELIMITER $$
USE `project2-moviedb`$$
CREATE DEFINER=`root`@`localhost` TRIGGER `appeared_in_BEFORE_INSERT`
BEFORE INSERT ON `appeared_in` FOR EACH ROW BEGIN
    if not exists (
        /* choose the star's other movie's type*/
        select *
        from appeared_in a, made_money b
        where
            a.star = new.star and
            a.movie = b.movie and
            b.day_opened < (select day_opened from made_money where movie
= new.movie) and
            b.category in ('Romantic', 'Comedy', 'Drama')
    )
    then
    update made_money m
    set m.category = 'Drama'
    where m.movie = new.movie ;
    end if;
END$$
DELIMITER ;
```

1.3. Question 3

```
DROP TRIGGER IF EXISTS `project2-moviedb`.`married_BEFORE_INSERT`;

DELIMITER $$
USE `project2-moviedb`$$
CREATE DEFINER = CURRENT_USER TRIGGER `project2-
moviedb`.`married_BEFORE_INSERT` BEFORE INSERT ON `married` FOR EACH ROW
BEGIN
    /* get the star's all divored day, repeating compare the new.day with
    * value in the new-get-temporary table to see if there exists a
violation
    * so we need a cursor, loop.
    **/
    declare done int default 0;
    declare divDate date;
    declare dateCursor cursor for (
    select divorced.day
    from (in_couple natural join married) left join divorced
    on married.couple_num = divorced.couple_num
    where (
        in_couple.couple_num = new.couple_num
    )
    );
    declare continue handler for not found set done = 1;
    open dateCursor;

    checkloop:repeat
    fetch dateCursor into divDate;
    if ( divDate = null or new.day < divDate ) then
        signal sqlstate '45000' /* an error occured, according to MySQL
manual */
        set message_text = 'cannot do this, because a star cannot be
married to multiple stars simultaneously';
        leave checkloop;
    end if;
    until done
    end repeat;

    close dateCursor;
END$$
DELIMITER ;
```

1.4. Question 4

```
DROP TRIGGER IF EXISTS `project2-moviedb`.`made_money_BEFORE_INSERT`;

DELIMITER $$
USE `project2-moviedb`$$
CREATE DEFINER = CURRENT_USER TRIGGER `project2-
moviedb`.`made_money_BEFORE_INSERT` BEFORE INSERT ON `made_money` FOR EACH
ROW FOLLOWS `movie_category_insert`
begin
        if ( new.category = 'Action' and new.how_much < 10000 ) then
                signal sqlstate '45000'
                set message_text ='An action movie need make no less than
$10,000.';
        elseif(new.category='Comedy' and new.how_much>1000000000) then
                signal sqlstate '45000'
                set message_text ='A Comedy cannot make more than
$1,000,000,000.';
        elseif(new.how_much<1000 or new.how_much>3000000000) then
                signal sqlstate'45000'
                set message_text ='A movie cannot make less than $1,000 or
larger than $3,000,000,000.';
        end if;
end

$$
DELIMITER ;
DROP TRIGGER IF EXISTS `project2-moviedb`.`made_money_BEFORE_UPDATE`;

DELIMITER $$
USE `project2-moviedb`$$
CREATE DEFINER = CURRENT_USER TRIGGER `project2-
```

```
moviedb`.`made_money_BEFORE_UPDATE` BEFORE UPDATE ON `made_money` FOR EACH
ROW FOLLOWS `movie_category_update`
begin
        if ( new.category = 'Action' and new.how_much < 10000 ) then
                signal sqlstate '45000'
                set message_text ='An action movie need make no less than
$10,000.';
        elseif(new.category='Comedy' and new.how_much>1000000000) then
                signal sqlstate '45000'
                set message_text ='A Comedy cannot make more than
$1,000,000,000.';
        elseif(new.how_much<1000 or new.how_much>3000000000) then
                signal sqlstate'45000'
                set message_text ='A movie cannot make less than $1,000 or
larger than $3,000,000,000.';
        end if;
end

$$
DELIMITER ;
```

1.5. Question 5

```
DROP TRIGGER IF EXISTS `project2-moviedb`.`divorced_BEFORE_INSERT`;

DELIMITER $$
USE `project2-moviedb`$$
CREATE DEFINER = CURRENT_USER TRIGGER `project2-
moviedb`.`divorced_BEFORE_INSERT` BEFORE INSERT ON `divorced` FOR EACH ROW
BEGIN
        declare dayDiv date ;
    select day into dayDiv from married where married.couple_num =
new.couple_num;
        if new.day < dayDiv
    then
    set new.day = dayDiv;
    end if;
END$$
DELIMITER ;
```

1.6. Question 6(Credit)

```
DROP TRIGGER IF EXISTS `project2-moviedb`.`made_money_BEFORE_INSERT_1`;

DELIMITER $$
USE `project2-moviedb`$$
CREATE DEFINER = `root`@`localhost` TRIGGER `project2-
```

```
moviedb`.`made_money_BEFORE_INSERT_1` BEFORE INSERT ON `made_money` FOR
EACH ROW FOLLOWS `made_money_BEFORE_INSERT`
begin
        insert into log_data values ( new.movie, new.category);
end
$$
DELIMITER ;
```

2. Observation

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

Movie	How_much	Day_opened	Category
'IRON MAN'	'1000000.00'	'2008-05-02'	'Action'

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

Movie	How_much	Day_opened	Category
'Fight Club'	'37023395.00'	'1999-10-15'	'Action'

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

Star	Movie
'Matt Damon'	'Bruce Almighty'

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

Time	Action	Response	Duration
22:26:06	insert into married values(1, 20150626)	Error Code: 1644. cannot do this, because a star cannot be married to multiple stars simultaneously	0.0073 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”). First insertion:

Time	Action	Response	Duration
22:29:40	insert into made_money values('Most Welcome', 8000, 20120707, 'Action')	Error Code: 1644. An action movie need make no less than $10,000.	0.00040 sec

Second insertion:

Movie	How_much	Day_opened	Category
'Speed'	'9000.00'	'2010-03-28'	'Comedy'

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

Time	Action	Response	Duration
22:32:31	insert into made_money value('Hangover', 1500000000, 20110305, 'Comedy')	Error Code: 1644. Cannot insert data because of the box office constraint.	0.00085 sec

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

Couple_num	Day
6	2005-06-25

3. LOG_DATA table

Movie	Category
'IRON MAN'	'Action'
'Speed'	'Comedy'