1. Course : EECS 495. Intro to Database Systems
2. Name : Sangrin Lee
3. Student ID : 2999428

‘

**Questions**:

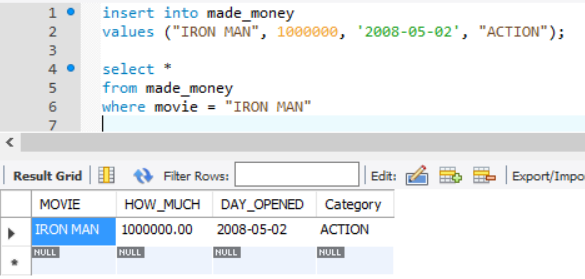
1. Enforce the constraint that categories of a movie must be either “Romantic”, “Comedy”, “Drama”, or “Action”. The default value for the Category field is “Action”. If a non-allowed value is inserted/updated, the category for that tuple must be changed to the default value.

[SQL Query]

|  |
| --- |
| DELIMITER $$  create trigger q1\_1  before insert on made\_money  for each row  begin  if new.category != "Romantic" and new.category != "Comedy" and new.category != "Drama" and new.category != "Action"  then  set new.category = "Action";  end if;  end;  $$  DELIMITER ;  DELIMITER $$  create trigger q1\_2  before update on made\_money  for each row  begin  if new.category != "Romantic" and new.category != "Comedy" and new.category != "Drama" and new.category != "Action"  then  set new.category = "Action";  end if;  end;  $$  DELIMITER ; |

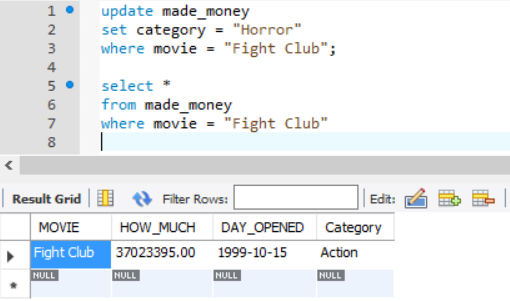
[Test - a]

a) Insert a new movie, with values (“IRON MAN”, 1000000, 2008-05-02, “ACTION”) in MADE\_MONEY table.



[Test – b]

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



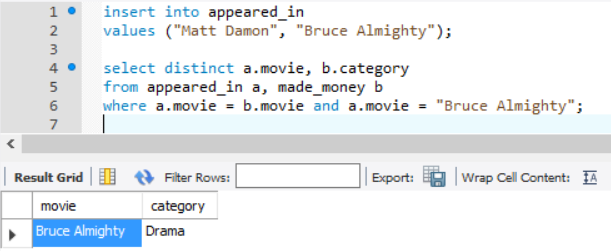
1. Enforce the following condition: A star can only be a part of a “Comedy” movie, only if he/she has performed in at least one “Romantic”, “Comedy”, or “Drama” movie previously. Upon insertion of a tuple violating this (e.g., a Comedy movie associated with a star who has previously done only “Action”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

[SQL Query]

|  |
| --- |
| DELIMITER $$  create trigger q2\_1  before insert on appeared\_in  for each row  begin  if(new.star not in  (select d.star from appeared\_in d, made\_money e  where d.movie = e.movie and  (e.category = "Romantic" or e.category = "Comedy" or e.category = "Drama")))  then  update made\_money f  set f.category = "Drama"  where f.movie = new.movie;  end if;  end;  $$  DELIMITER ;  DELIMITER $$  create trigger q2\_2  before update on appeared\_in  for each row  begin  if(new.star not in  (select d.star from appeared\_in d, made\_money e  where d.movie = e.movie and  (e.category = "Romantic" or e.category = "Comedy" or e.category = "Drama")))  then  update made\_money f  set f.category = "Drama"  where f.movie = new.movie;  end if;  end;  $$  DELIMITER ; |

[Test - c]

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



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

(Assumption : There’s no multiple day of marriage and divorce for inserting and updating tuples.)

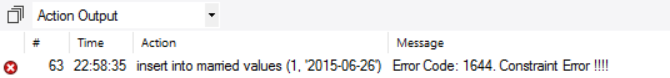
[SQL Query]

|  |
| --- |
| DELIMITER $$  create trigger q3\_1  before insert on married  for each row  begin  if (  ((select b.star from in\_couple b where b.couple\_num = new.couple\_num limit 1)  in (select a.star from in\_couple a, married c where a.couple\_num = c.couple\_num)  and  (select e.star from in\_couple e where e.couple\_num = new.couple\_num limit 1)  in (select d.star from in\_couple d, divorced f where d.couple\_num = f.couple\_num)  and  (new.day >  (select a.day from married a, in\_couple b where a.couple\_num = b.couple\_num and b.star =  (select c.star from in\_couple c where c.couple\_num = new.couple\_num limit 1)))  and  (new.day <  (select a.day from divorced a, in\_couple b where a.couple\_num = b.couple\_num and b.star =  (select c.star from in\_couple c where c.couple\_num = new.couple\_num limit 1))))  or  ((select b.star from in\_couple b where b.couple\_num = new.couple\_num limit 1)  in (select a.star from in\_couple a, married c where a.couple\_num = c.couple\_num)  and  (select e.star from in\_couple e where e.couple\_num = new.couple\_num limit 1)  not in (select d.star from in\_couple d, divorced f where d.couple\_num = f.couple\_num)  and  (date(new.day) >  (select a.day from married a, in\_couple b where a.couple\_num = b.couple\_num and b.star =  (select c.star from in\_couple c where c.couple\_num = new.couple\_num limit 1))))  or  ((select b.star from in\_couple b where b.couple\_num = new.couple\_num limit 1, 1)  in (select a.star from in\_couple a, married c where a.couple\_num = c.couple\_num)  and  (select e.star from in\_couple e where e.couple\_num = new.couple\_num limit 1, 1)  in (select d.star from in\_couple d, divorced f where d.couple\_num = f.couple\_num)  and  (new.day >  (select a.day from married a, in\_couple b where a.couple\_num = b.couple\_num and b.star =  (select c.star from in\_couple c where c.couple\_num = new.couple\_num limit 1, 1)))  and  (new.day <  (select a.day from divorced a, in\_couple b where a.couple\_num = b.couple\_num and b.star =  (select c.star from in\_couple c where c.couple\_num = new.couple\_num limit 1, 1))))  or  ((select b.star from in\_couple b where b.couple\_num = new.couple\_num limit 1, 1)  in (select a.star from in\_couple a, married c where a.couple\_num = c.couple\_num)  and  (select e.star from in\_couple e where e.couple\_num = new.couple\_num limit 1, 1)  not in (select d.star from in\_couple d, divorced f where d.couple\_num = f.couple\_num)  and  (date(new.day) >  (select a.day from married a, in\_couple b where a.couple\_num = b.couple\_num and b.star =  (select c.star from in\_couple c where c.couple\_num = new.couple\_num limit 1, 1))))  )then  SIGNAL SQLSTATE 'ERROR' SET MESSAGE\_TEXT = 'Constraint Error !!!!';  end if;  end;  $$  DELIMITER ;  DELIMITER $$  create trigger q3\_2  before update on married  for each row  begin  if (  ((select b.star from in\_couple b where b.couple\_num = new.couple\_num limit 1)  in (select a.star from in\_couple a, married c where a.couple\_num = c.couple\_num)  and  (select e.star from in\_couple e where e.couple\_num = new.couple\_num limit 1)  in (select d.star from in\_couple d, divorced f where d.couple\_num = f.couple\_num)  and  (new.day >  (select a.day from married a, in\_couple b where a.couple\_num = b.couple\_num and b.star =  (select c.star from in\_couple c where c.couple\_num = new.couple\_num limit 1)))  and  (new.day <  (select a.day from divorced a, in\_couple b where a.couple\_num = b.couple\_num and b.star =  (select c.star from in\_couple c where c.couple\_num = new.couple\_num limit 1))))  or  ((select b.star from in\_couple b where b.couple\_num = new.couple\_num limit 1)  in (select a.star from in\_couple a, married c where a.couple\_num = c.couple\_num)  and  (select e.star from in\_couple e where e.couple\_num = new.couple\_num limit 1)  not in (select d.star from in\_couple d, divorced f where d.couple\_num = f.couple\_num)  and  (date(new.day) >  (select a.day from married a, in\_couple b where a.couple\_num = b.couple\_num and b.star =  (select c.star from in\_couple c where c.couple\_num = new.couple\_num limit 1))))  or  ((select b.star from in\_couple b where b.couple\_num = new.couple\_num limit 1, 1)  in (select a.star from in\_couple a, married c where a.couple\_num = c.couple\_num)  and  (select e.star from in\_couple e where e.couple\_num = new.couple\_num limit 1, 1)  in (select d.star from in\_couple d, divorced f where d.couple\_num = f.couple\_num)  and  (new.day >  (select a.day from married a, in\_couple b where a.couple\_num = b.couple\_num and b.star =  (select c.star from in\_couple c where c.couple\_num = new.couple\_num limit 1, 1)))  and  (new.day <  (select a.day from divorced a, in\_couple b where a.couple\_num = b.couple\_num and b.star =  (select c.star from in\_couple c where c.couple\_num = new.couple\_num limit 1, 1))))  or  ((select b.star from in\_couple b where b.couple\_num = new.couple\_num limit 1, 1)  in (select a.star from in\_couple a, married c where a.couple\_num = c.couple\_num)  and  (select e.star from in\_couple e where e.couple\_num = new.couple\_num limit 1, 1)  not in (select d.star from in\_couple d, divorced f where d.couple\_num = f.couple\_num)  and  (date(new.day) >  (select a.day from married a, in\_couple b where a.couple\_num = b.couple\_num and b.star =  (select c.star from in\_couple c where c.couple\_num = new.couple\_num limit 1, 1))))  )then  SIGNAL SQLSTATE 'ERROR' SET MESSAGE\_TEXT = 'Constraint Error !!!!';  end if;  end;  $$  DELIMITER ; |

[Test - d]

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





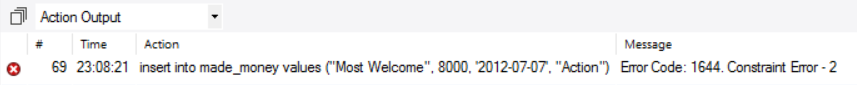
1. Enforce that, a movie must make at least $1,000 in the box office, and cannot make more than 3 billion ($3,000,000,000) in the box office. Also, if a movie category is “Action”, then it should make at least $10,000, and if category is “Comedy”, it cannot make more than $1,000,000,000.

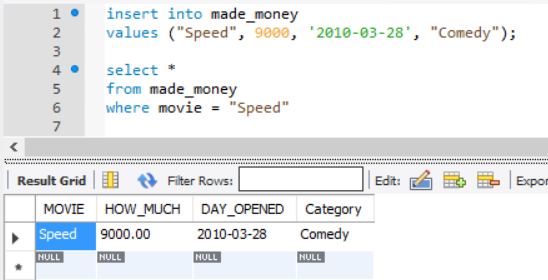
[SQL Query]

|  |
| --- |
| DELIMITER $$  create trigger q4\_1  before insert on made\_money  for each row  begin  if (new.how\_much < 1000 and new.how\_much > 3000000000)  then  SIGNAL SQLSTATE 'ERROR' SET MESSAGE\_TEXT = 'Constraint Error - 1';  else if (new.category = "Action" and new.how\_much < 10000)  then  SIGNAL SQLSTATE 'ERROR' SET MESSAGE\_TEXT = 'Constraint Error - 2';  else if (new.category = "Comedy" and new.how\_much > 1000000000)  then  SIGNAL SQLSTATE 'ERROR' SET MESSAGE\_TEXT = 'Constraint Error - 3';  end if;  end if;  end if;  end;  $$  DELIMITER ;  DELIMITER $$  create trigger q4\_2  before update on made\_money  for each row  begin  if (new.how\_much < 1000 and new.how\_much > 3000000000)  then  SIGNAL SQLSTATE 'ERROR' SET MESSAGE\_TEXT = 'Constraint Error - 1';  else if (new.category = "Action" and new.how\_much < 10000)  then  SIGNAL SQLSTATE 'ERROR' SET MESSAGE\_TEXT = 'Constraint Error - 2';  else if (new.category = "Comedy" and new.how\_much > 1000000000)  then  SIGNAL SQLSTATE 'ERROR' SET MESSAGE\_TEXT = 'Constraint Error - 3';  end if;  end if;  end if;  end;  $$  DELIMITER ; |

[Test - e]

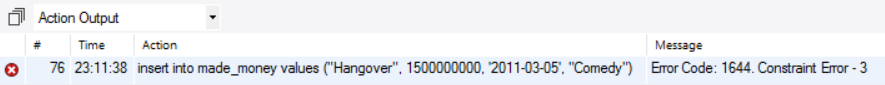
e) Insert two new tuples in MADE\_MONEY, having values (“Most Welcome”, 8000, 2012-07-07, “Action”) and (“Speed”, 9000, 2010-03-28, “Comedy”).





[Test - f]

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



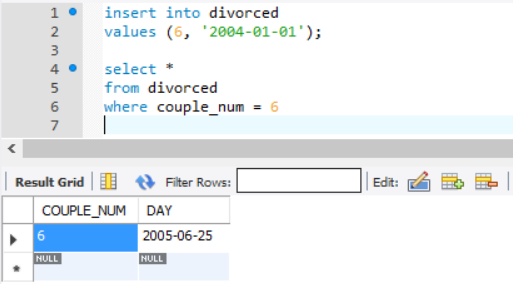
1. Using a trigger, ensure that the divorce date of a couple is at least the same or after their marriage date. If this is violated, set the divorce date to be the same as the marriage date.

[SQL Query]

|  |
| --- |
| DELIMITER $$  create trigger q5\_1  before insert on divorced  for each row  begin  if (date(NEW.day) < (select date(b.day) from married b  where b.couple\_num = new.couple\_num))  then  set new.day = (select date(b.day) from married b where b.couple\_num = new.couple\_num);  end if;  end;  $$  DELIMITER ;  DELIMITER $$  create trigger q5\_2  before update on divorced  for each row  begin  if (date(NEW.day) < (select date(b.day) from married b  where b.couple\_num = new.couple\_num))  then  set new.day = (select date(b.day) from married b where b.couple\_num = new.couple\_num);  end if;  end;  $$  DELIMITER ; |

[Test - g]

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



1. 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.

[SQL Query]

|  |
| --- |
| DELIMITER $$  create trigger q6  after insert on made\_money  for each row  begin  insert into log\_data  values (new.movie, new.category);  end;  $$  DELIMITER ; |

[Test & Output]

