```
/*Qui, à l'ajout de costumes dans la table Costume, vérifie si les numéros de
stylistes concernés existent dans la table Styliste. Si ce n'est pas le cas annuler
l'opération d'ajout*/
delimiter $$
create trigger Tr1 before insert on costume
for EACH ROW
BEGIN
set @NumSt=new.NumStyliste;
/*
if @numst not in(select numStyliste from styliste )THEN ....end if

set @nbr=(select count(*) from styliste where numStyliste=@numSt);
if  @nbr>0 then .... end if;
*/
if not exists (select * from styliste where numStyliste=@numSt) THEN
          SIGNAL SQLSTATE '45000'
   set MESSAGE_TEXT='Ce styliste n''existe pas dans la base de données';
end if;
end$$
```

```
/*Qui, à la suppression de costumes, vérifie si des notes leur ont été attribuées.
Si c'est le cas empêcher la suppression
*/
delimiter $$
create TRIGGER TR2 before DELETE on costume
for each ROW
begin
/*récupérer le numero de costume à supprmier*/
set @NumCos=old.NumCostume;
/*if EXISTS (select * from noteJury where NumCostume=@numCos )then .. end IF*/
if @NumCos in (select NumCostume FROM noteJury) THEN
          SIGNAL SQLSTATE '40000'
   set MESSAGE_TEXT ='Ce costume est déjà noté , On ne peut pas le supprimer!!!';
end if;

end$$
```

```
/*Qui, à l'affectation de notes à des costumes dans la table NotesJury, vérifie si
les costumes et les membres de jury existent et si les notes attribuées sont
comprises entre 0 et 20*/
DELIMITER $$
create TRIGGER TR3 before insert on notejury
for each ROW
BEGIN
/*récupérer le num de costume à noter*/
set @numCos=new.NumCostume;
if @numCos not in (select NumCostume from costume) then
    signal SQLSTATE '45000'
    set MESSAGE_TEXT='Ce costume n''existe pas dans la base de données';
end if;
/*récupérer le num de membre du jury */
set @numJury=new.NumMembreJury;
if @numJury not in (select NumMembreJury from mebmreJury)  THEN
```

```
            signal SQLSTATE '45000'
   set MESSAGE_TEXT='Ce membre jury n''existe pas dans la base de données';
end if;

/*récupérer la note à attribuer*/
set @note= new.NoteAttribuee;
if @note<0 or @note>20 THEN
            signal SQLSTATE '45000'
   set MESSAGE_TEXT='La note doitêtre entre  0 et 20';
end if;

end$$
```

```
delimiter $$
drop TRIGGER if exists  TR3;$$
create TRIGGER if not exists  TR3 before insert on notejury
for each ROW
BEGIN
set @erreur='';
set @valide=1;
/*récupérer le num de costume à noter*/
set @numCos=new.NumCostume;
if @numCos not in (select NumCostume from costume) then
     set @erreur='Ce costume n''existe pas dans la base de données';
     set @valide=-1;
end if;
/*récupérer le num de membre du jury */
set @numJury=new.NumMembreJury;
if @numJury not in (select NumMembreJury from membreJury)  THEN
set @valide=-1;
   set @erreur=concat(@erreur,'\n','Ce membre jury n''existe pas dans la base de données');
end if;

/*récupérer la note à attribuer*/
set @note= new.NoteAttribuee;
if @note<0 or @note>20 THEN
            set @valide=-1;
   set @erreur=concat(@erreur,'\n','La note doit être entre  0 et 20');
end if;
if  @valide= -1 then
            signal SQLSTATE '45000'
          set MESSAGE_TEXT=@erreur;
end if;
end$$
```

```
/*Qui à l'ajout de membres jury, cherche si leurs fonctions existent dans la table
Fonction si ce n'est pas le cas il les rajoute.*/
delimiter $$
drop trigger if exists TR4;$$
create trigger if not exists TR4 before insert on membrejury
for each ROW
BEGIN
```

```
/*récupérer le fonction du mebmreJury à ajoute*/
set @f=new.Fonction;
if not exists ( select * from fonction where fonction like @f) then

   insert into fonction values(@f);
end if;

end$$
```