Applied Database

1. trigger customer_create_date, payment_date, rental_date memiliki fungsi yang sama yaitu untuk menambahkan tanggal sebelum melakukan insert.
2. upd_film memiliki fungsi yaitu untuk mengupdate judul film dan mengubah deskripsi dari sebuah film.
3.
```
delimiter $;
CREATE trigger tLastUpdateRental;
BEFORE UPDATE ON rental for each row
BEGIN
        set new.last_update date = now();
END
delimiter ;
```

4.
```
ALTER TABLE customer ADD total_bulan_pinjam decimal(4,2);
DROP PROCEDURE IF EXISTS uspUpdateTotalPinjam;
DELIMITER $;
CREATE PROCEDURE uspUpdateTotalPinjam()
BEGIN
        DECLARE done int default 0;
        DECLARE totalLamaPinjam float default 0.0;
        DECLARE curCustomerPinjam CURSOR FOR select customer_id from
customer;
   DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
   DECLARE kodeCustomer int;

   OPEN curCustomerPinjam;
   FETCH curCustomerPinjam into kodeCustomer;



END

DELIMITER ;
```

5.
```
DROP PROCEDURE IF EXISTS uspSimpanPeminjaman;
DELIMITER $;
CREATE PROCEDURE uspSimpanPeminjaman(in cityName varchar(45), in
filmTitle varchar(45), in idCustomer int(11),
in idStaff int(11), out hasil varchar(45))
BEGIN
        DECLARE kodeFilm int;
   DECLARE stok int;
   DECLARE jumlahPinjam int;
   DECLARE alamat_id int;
   DECLARE kodeInventory int;
   select address_id into alamat_id from address a inner join city c on a.city_id =
c.city_id where c.city = cityName;
   select film_id into kodeFilm from film where title = filmTitle;
         select count(inventory_id) into stok from inventory where film_id = kodeFilm;


   select count(r.rental_id) into jumlahPinjam from rental as r
   inner join inventory as i on r.inventory_id = i.inventory_id where i.film_id =
kodeFilm and r.return_date = null;

   IF(jumlahPinjam != stok) then
                select i.inventory_id into kodeInventory from inventory i
      inner join rental r on i.inventory_id = r.inventory_id where i.film_id = kodeFilm
      and i.store_id = alamat_id limit 1;

      insert into rental(rental_date, inventory_id, customer_id, staff_id, last_update)
values (now(), kodeInventory+1, idCustomer, idStaff, now());
      set hasil = 'BERHASIL MENYEWA';
         END IF;
END $;
DELIMITER ;

call uspSimpanPeminjaman('Lethbrigde', 'ACADEMY DINOSAUR', 1, 2, @hasil);
select @hasil;
```

6.
```
DROP PROCEDURE IF EXISTS uspTambahFilm;
DELIMITER $;
CREATE PROCEDURE uspTambahFilm(in judulFilm varchar(45), in deskripsi
varchar(255), in tanggal_rilis int,
 in bahasa int, in lama_rental int, in rate_rental float(0,2), in durasi_film int, in
biaya_ganti_rugi float(0,2)
 , in rating_usia varchar(45), in bonus_features varchar(120), in last_update datetime)
 BEGIN
         INSERT INTO film(title, description, release_year, language_id,
original_language, rental_duration, rental_rate,
   length, replacement_cost, rating, special_features, last_update)
VALUES(judulFilm, deskripsi, tanggal_rilis,
   bahasa, lama_rental, rate_rental, durasi_film, biaya_ganti_rugi, rating_usia,
bonus_features, last_update);
END $;
DELIMITER ;
```