Assignment No.8

```
delimiter //

CREATE TRIGGER calculate_price
BEFORE insert
on selects FOR each row
BEGIN

  SET  NEW.price=NEW.quantity*(select price from product
where NEW.product_id=product.product_id);
  UPDATE orders SET total_price=total_price+NEW.price;

END //

delimiter ;

delimiter //

create TRIGGER capitalize_name
before insert
on customer for each row
begin

  set NEW.fname= CONCAT((UPPER(SUBSTRING(NEW.fname, 1,
1))),SUBSTR(NEW.fname, 2));
  set NEW.lname= CONCAT((UPPER(SUBSTRING(NEW.lname, 1,
1))),SUBSTR(NEW.lname, 2));

 end //

delimiter ;

delimiter //

CREATE TRIGGER check_quantity
BEFORE insert
on selects FOR each row
BEGIN

  IF NEW.quantity<=0 THEN
    SIGNAL SQLSTATE '80000'
        SET MESSAGE_TEXT = 'Quantity should be greater than
0';
  END IF;
```

```
END //

delimiter ;

delimiter //

CREATE TRIGGER customerdeletelog
BEFORE delete
on customer FOR each row
BEGIN

  insert into
customer_log(cust_id,oldfname,newfname,oldlname,newlname,oldz
ipcode,newzipcode,oldphone_no,newphone_no,user,tstamp)
values(OLD.cust_id,OLD.fname,NULL,OLD.lname,NULL,OLD.zipcode,
NULL,OLD.phone_no,NULL,current_user(),current_timestamp());

END //

delimiter ;

delimiter //

CREATE TRIGGER customerinsertlog
AFTER insert
on customer FOR each row
BEGIN

  insert into
customer_log(cust_id,oldfname,newfname,oldlname,newlname,oldz
ipcode,newzipcode,oldphone_no,newphone_no,user,tstamp)
values(NEW.cust_id,NULL,NEW.fname,NULL,NEW.lname,NULL,NEW.zip
code,NULL,NEW.phone_no,current_user(),current_timestamp());

END //

delimiter ;

delimiter //

CREATE TRIGGER customerupdatelog
AFTER update
on customer FOR each row
BEGIN

  insert into
customer_log(cust_id,oldfname,newfname,oldlname,newlname,oldz
ipcode,newzipcode,oldphone_no,newphone_no,user,tstamp)
```

```
values(OLD.cust_id,OLD.fname,NEW.fname,OLD.lname,NEW.lname,OL
D.zipcode,NEW.zipcode,OLD.phone_no,NEW.phone_no,current_user(
),current_timestamp());

END //

delimiter ;

delimiter //

CREATE TRIGGER orderdeletelog
BEFORE delete
on selects FOR each row
BEGIN

  insert into
order_log(order_id,product_id,oldquantity,newquantity,oldpric
e,newprice,user,tstamp)
values(OLD.order_id,OLD.product_id,OLD.quantity,NULL,OLD.pric
e,NULL,current_user(),current_timestamp());

END //

delimiter ;

delimiter //

CREATE TRIGGER orderinsertlog
AFTER INSERT
on selects FOR each row
BEGIN

  insert into
order_log(order_id,product_id,oldquantity,newquantity,oldpric
e,newprice,user,tstamp)
values(NEW.order_id,NEW.product_id,NULL,NEW.quantity,NULL,NEW
.price,current_user(),current_timestamp());

END //

delimiter ;

delimiter //

CREATE TRIGGER orderupdatelog
AFTER update
on selects FOR each row
BEGIN
```

```
  insert into
order_log(order_id,product_id,oldquantity,newquantity,oldpric
e,newprice,user,tstamp)
values(OLD.order_id,OLD.product_id,OLD.quantity,NEW.quantity,
OLD.price,NEW.price,current_user(),current_timestamp());

END //

delimiter ;

delimiter //

CREATE TRIGGER update_stock
BEFORE insert
on selects FOR each row
BEGIN

  IF (NEW.quantity>(select stock from product where
NEW.product_id=product.product_id)) THEN
    SIGNAL SQLSTATE '80000'
        SET MESSAGE_TEXT = 'Order exceeding stock';
  ELSE
    update product set stock=stock-NEW.quantity where
NEW.product_id=product.product_id;
  END IF;

END //

delimiter ;
```