# Functions

**1. Calculates due premium amount for lapsed as well as active policies.**

create function calculate\_due\_premium(policynum int(11)) returns int(11) DETERMINISTIC BEGIN declare due int(11);   
declare t int(11);  
select TIMESTAMPDIFF(MONTH, dueDate, CURDATE()-1) into t from policy where policyNumber = policynum;   
IF t < 0 THEN  
set t = 0;  
END IF;  
select ((1 + FLOOR(t/policy.payment\_interval))\*policy.premium) into due from policy where policy.policyNumber = policynum;  
return (due);  
END//

**2. Calculate commission to be paid to the agent upon premium payment.**

create function calculate\_commission(policynum int(11)) returns int(11) DETERMINISTIC BEGIN declare com int(11);   
declare t int(11);  
select TIMESTAMPDIFF(MONTH, dueDate, CURDATE()-1) into t from policy where policyNumber = policynum;   
IF t < 0 THEN  
set t = 0;  
END IF;  
select ((1 +FLOOR( t/policy.payment\_interval))\* policy.premium\*plans.comRate\*0.01) into com from policy, plans where policy.planID = plans.planID and policy.policyNumber = policynum;  
return (com);  
END//

**3. Calculates the late fee applicable on paying premium after due date at the following rates:**

* **Up to 1 month from due date: 0% (Grace Period)**
* **After 1 month till next due date: 5%**
* **After 2 missed due dates: 10%**

create function calculate\_late\_fee(policynum int(11)) returns int(11) DETERMINISTIC BEGIN declare fee int(11);   
declare t int(11);  
select TIMESTAMPDIFF(MONTH, dueDate, CURDATE()-1) into t from policy where policyNumber = policynum;   
if t > 0 THEN select ((1 + FLOOR(t/policy.payment\_interval))\*policy.premium\* t/240) into fee from policy where policy.policyNumber=policynum;   
else   
set fee = 0; end if; return (fee); END//

# Procedures

**1. Update premium due date for policy according to premium payment interval. Set due date = NULL for paid up policies.**

create procedure update\_due\_date(policynum int)  
BEGIN   
select (TIMESTAMPDIFF(MONTH, dueDate, CURDATE()-1)) into @t from policy where policyNumber = policynum;   
IF @t < 0 THEN  
set @t = 0;  
END IF;  
update policy set dueDate = DATE\_ADD(dueDate, INTERVAL (1+FLOOR(@t/payment\_interval))\*payment\_interval MONTH) where policyNumber = policynum;  
update policy set dueDate = NULL where dueDate = (select DATE\_ADD(startDate, INTERVAL duration YEAR) from plans where policy.planID = plans.planID) and policyNumber = policynum;  
END//

**2. Update status of lapsed and expired policies.**

create procedure update\_status() BEGIN  
update policy set status = 'Lapsed' where TIMESTAMPDIFF(MONTH, dueDate, CURDATE()-1) >= 1;  
update policy set status = 'Expired', dueDate = NULL where policyNumber = (select policyNumber from plans where CURDATE()-1 =DATE\_ADD(startDate, INTERVAL duration YEAR) and policy.planID = plans.planID and (status='Active' or status='Expired'));   
END//

# Views

**1.**

create view agent\_commission   
as   
select agent.agentID, agentName, sum(commission) as commission from policy, agent where policy.agentID=agent.agentID group by agent.agentID;

**2.**

create view reminders   
as   
select \* from policies where DATEDIFF(dueDate, CURDATE()) <=10 and DATEDIFF(dueDate, CURDATE()) >= -10;

# Triggers

**1. Pay commission to the agent after premium payment**

create trigger pay\_commission AFTER INSERT on payment for each row BEGIN update policy set commission = (commission + calculate\_commission(new.policyNumber)) where policyNumber = new.policyNumber;   
END//

**2. Update due date after premium payment and re-activate lapsed policies**

create trigger after\_payment AFTER INSERT on payment for each row BEGIN call update\_due\_date(new.policynumber);  
update policy set status = 'Active' where DATEDIFF(dueDate, CURDATE()-1) >= 1 and policyNumber = new.policyNumber;   
END//

**3. Update policy status upon filing of new claim**

create trigger new\_claim  
AFTER INSERT on claim  
for each row   
BEGIN  
update policy set status = 'Claim Filed' where policy.policyNumber = new.policyNumber;  
END//

**4. Update policy status upon claim approval/rejection**

create trigger update\_claim  
AFTER UPDATE on claim  
for each row   
BEGIN  
IF new.status = 'Under Process' THEN  
update policy set status = 'Claim Filed' where policy.policyNumber = new.policyNumber;  
ELSEIF new.status = 'Claim Approved' THEN  
update policy set status = 'Claimed', dueDate = NULL where policy.policyNumber = new.policyNumber;  
ELSEIF  
update policy set status = 'Closed', dueDate = NULL where policy.policyNumber = new.policyNumber;  
END IF;  
END//