**Problem statement**

The database is for a general Insurance firm, comprising of the different components of Insurance. The database is currently focused on only Motor insurance.

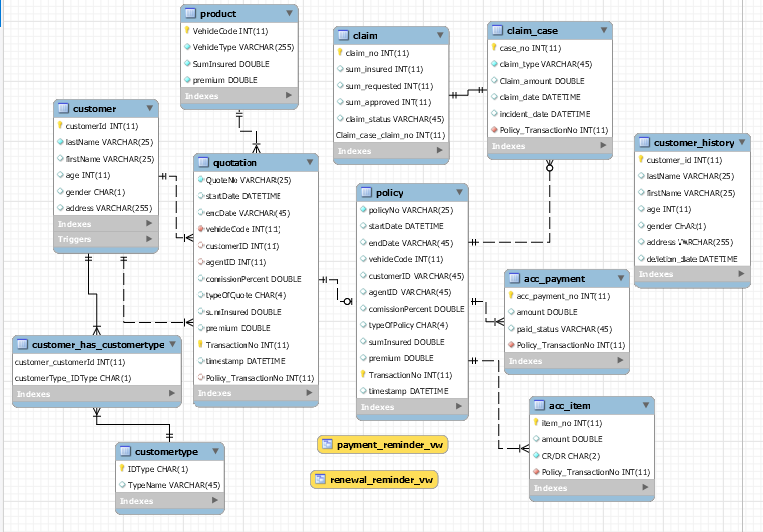
It covers the various steps involved from adding a customer to assigning it roles till his policy expires with the insurance firm.

**Approach Followed**

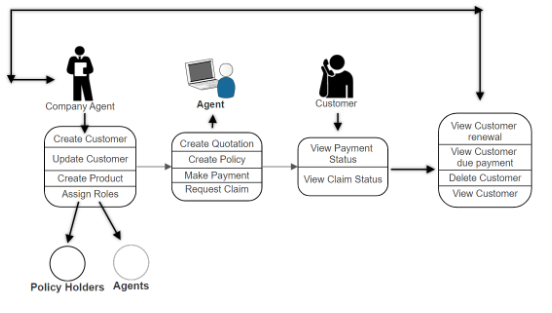
The Insurance firm will comprise of the following entities:

1. Create Customer: Allowing the Insurance firm to add, update and delete records of customers.
2. Assign Roles: Assign roles to them of either the Policy Holder or an Agent.
3. Create Quotation: Create and update Quotations and later convert them into policies.
4. Convert to Policies: The finalised quotation agreed by both the parties can later be converted into Policies with the accounting entries passed to the system.
5. Payments: Allowing customers to make payments for their policy in instalments and in full and getting a refund in scenarios of being overpaid.
6. Claim: The customer then also can request for claims against their respective policies which is found applicable to the motor insurance firm will be approved/rejected accordingly.
7. Analysis: It is carried out by the insurance firm to be updated with the payment status of each employee and track them for renewal if required.

**ER Diagram**



**Use case diagram**



**Definitions**

**DDL:**

SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;

SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;

SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION';

-- -----------------------------------------------------

-- Database motor\_insurance

-- -----------------------------------------------------

DROP DATABASE IF EXISTS motor\_insurance;

CREATE DATABASE IF NOT EXISTS motor\_insurance;

USE motor\_insurance;

-- -----------------------------------------------------

-- Table `motor\_insurance`.`customer`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `motor\_insurance`.`customer` ;

CREATE TABLE IF NOT EXISTS `motor\_insurance`.`customer` (

`customerId` INT NOT NULL AUTO\_INCREMENT,

`lastName` VARCHAR(25) NOT NULL,

`firstName` VARCHAR(25) NULL,

`age` INT NULL,

`gender` CHAR NULL,

`address` VARCHAR(255) NULL,

PRIMARY KEY (`customerId`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `motor\_insurance`.`Product`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `motor\_insurance`.`Product` ;

CREATE TABLE IF NOT EXISTS `motor\_insurance`.`Product` (

`VehicleCode` INT NOT NULL,

`VehicleType` VARCHAR(255) NOT NULL,

`SumInsured` DOUBLE NOT NULL,

`premium` DOUBLE NOT NULL,

PRIMARY KEY (`VehicleCode`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `motor\_insurance`.`Policy`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `motor\_insurance`.`Policy` ;

CREATE TABLE IF NOT EXISTS `motor\_insurance`.`Policy` (

`policyNo` VARCHAR(25) NOT NULL DEFAULT 'NA',

`startDate` DATETIME NULL,

`endDate` VARCHAR(45) NULL,

`vehicleCode` INT NULL,

`customerID` VARCHAR(45) NULL,

`agentID` VARCHAR(45) NULL,

`comissionPercent` DOUBLE NULL,

`typeOfPolicy` CHAR(4) NULL,

`sumInsured` DOUBLE NULL,

`premium` DOUBLE NULL,

`TransactionNo` INT NOT NULL AUTO\_INCREMENT,

`timestamp` DATETIME NULL,

PRIMARY KEY (`TransactionNo`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `motor\_insurance`.`acc\_item`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `motor\_insurance`.`acc\_item` ;

CREATE TABLE IF NOT EXISTS `motor\_insurance`.`acc\_item` (

`item\_no` INT NOT NULL AUTO\_INCREMENT,

`amount` DOUBLE NULL,

`CR/DR` CHAR(2) NOT NULL,

`Policy\_TransactionNo` INT NOT NULL DEFAULT 0,

PRIMARY KEY (`item\_no`),

INDEX `fk\_acc\_item\_Policy1\_idx` (`Policy\_TransactionNo` ASC) VISIBLE,

CONSTRAINT `fk\_acc\_item\_Policy1`

FOREIGN KEY (`Policy\_TransactionNo`)

REFERENCES `motor\_insurance`.`Policy` (`TransactionNo`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `motor\_insurance`.`customerType`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `motor\_insurance`.`customerType` ;

CREATE TABLE IF NOT EXISTS `motor\_insurance`.`customerType` (

`IDType` CHAR NOT NULL,

`TypeName` VARCHAR(45) NULL,

PRIMARY KEY (`IDType`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `motor\_insurance`.`customer\_has\_customerType`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `motor\_insurance`.`customer\_has\_customerType` ;

CREATE TABLE IF NOT EXISTS `motor\_insurance`.`customer\_has\_customerType` (

`customer\_customerId` INT NOT NULL,

`customerType\_IDType` CHAR NOT NULL,

PRIMARY KEY (`customer\_customerId`, `customerType\_IDType`),

INDEX `fk\_customer\_has\_customerType\_customerType1\_idx` (`customerType\_IDType` ASC) VISIBLE,

INDEX `fk\_customer\_has\_customerType\_customer1\_idx` (`customer\_customerId` ASC) VISIBLE,

CONSTRAINT `fk\_customer\_has\_customerType\_customer1`

FOREIGN KEY (`customer\_customerId`)

REFERENCES `motor\_insurance`.`customer` (`customerId`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_customer\_has\_customerType\_customerType1`

FOREIGN KEY (`customerType\_IDType`)

REFERENCES `motor\_insurance`.`customerType` (`IDType`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `motor\_insurance`.`Quotation`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `motor\_insurance`.`Quotation` ;

CREATE TABLE IF NOT EXISTS `motor\_insurance`.`Quotation` (

`QuoteNo` VARCHAR(25) NOT NULL DEFAULT 'NA',

`startDate` DATETIME NULL,

`endDate` VARCHAR(45) NULL,

`vehicleCode` INT NOT NULL,

`customerID` INT NULL,

`agentID` INT NULL,

`comissionPercent` DOUBLE NULL,

`typeOfQuote` CHAR(4) NULL,

`sumInsured` DOUBLE NULL,

`premium` DOUBLE NULL,

`TransactionNo` INT NOT NULL AUTO\_INCREMENT,

`timestamp` DATETIME NULL,

`Policy\_TransactionNo` INT NULL,

PRIMARY KEY (`TransactionNo`),

INDEX `fk\_Policy\_customer0\_idx` (`customerID` ASC) VISIBLE,

INDEX `fk\_Policy\_customer1\_idx` (`agentID` ASC) VISIBLE,

INDEX `fk\_Quotation\_Policy1\_idx` (`Policy\_TransactionNo` ASC) VISIBLE,

INDEX `fk\_Policy\_Product10\_idx` (`vehicleCode` ASC) VISIBLE,

CONSTRAINT `fk\_Policy\_customer0`

FOREIGN KEY (`customerID`)

REFERENCES `motor\_insurance`.`customer` (`customerId`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_Policy\_customer1`

FOREIGN KEY (`agentID`)

REFERENCES `motor\_insurance`.`customer` (`customerId`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_Policy\_Product10`

FOREIGN KEY (`vehicleCode`)

REFERENCES `motor\_insurance`.`Product` (`VehicleCode`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_Quotation\_Policy1`

FOREIGN KEY (`Policy\_TransactionNo`)

REFERENCES `motor\_insurance`.`Policy` (`TransactionNo`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `motor\_insurance`.`Claim\_case`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `motor\_insurance`.`Claim\_case` ;

CREATE TABLE IF NOT EXISTS `motor\_insurance`.`Claim\_case` (

`case\_no` INT NOT NULL AUTO\_INCREMENT,

`claim\_type` VARCHAR(45) NOT NULL,

`Claim\_amount` DOUBLE NULL,

`claim\_date` DATETIME NULL,

`incident\_date` DATETIME NULL,

`Policy\_TransactionNo` INT NOT NULL,

PRIMARY KEY (`case\_no`),

INDEX `fk\_Claim\_Policy1\_idx` (`Policy\_TransactionNo` ASC) VISIBLE,

CONSTRAINT `fk\_Claim\_Policy1`

FOREIGN KEY (`Policy\_TransactionNo`)

REFERENCES `motor\_insurance`.`Policy` (`TransactionNo`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `motor\_insurance`.`claim`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `motor\_insurance`.`claim` ;

CREATE TABLE IF NOT EXISTS `motor\_insurance`.`claim` (

`claim\_no` INT NOT NULL AUTO\_INCREMENT,

`sum\_insured` INT NULL,

`sum\_requested` INT NULL,

`sum\_approved` INT NULL,

`claim\_status` VARCHAR(45) NULL,

`Claim\_case\_claim\_no` INT NOT NULL,

PRIMARY KEY (`claim\_no`, `Claim\_case\_claim\_no`),

INDEX `fk\_claim\_Claim\_case1\_idx` (`Claim\_case\_claim\_no` ASC) VISIBLE,

CONSTRAINT `fk\_claim\_Claim\_case1`

FOREIGN KEY (`Claim\_case\_claim\_no`)

REFERENCES `motor\_insurance`.`Claim\_case` (`case\_no`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `motor\_insurance`.`acc\_payment`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `motor\_insurance`.`acc\_payment` ;

CREATE TABLE IF NOT EXISTS `motor\_insurance`.`acc\_payment` (

`acc\_payment\_no` INT NOT NULL AUTO\_INCREMENT,

`amount` DOUBLE NULL,

`paid\_status` VARCHAR(45) NULL,

`Policy\_TransactionNo` INT NOT NULL,

PRIMARY KEY (`acc\_payment\_no`),

INDEX `fk\_acc\_payment\_Policy1\_idx` (`Policy\_TransactionNo` ASC) VISIBLE,

CONSTRAINT `fk\_acc\_payment\_Policy1`

FOREIGN KEY (`Policy\_TransactionNo`)

REFERENCES `motor\_insurance`.`Policy` (`TransactionNo`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `motor\_insurance`.`customer\_history`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `motor\_insurance`.`customer\_history` ;

CREATE TABLE IF NOT EXISTS `motor\_insurance`.`customer\_history` (

`customer\_id` INT NOT NULL,

`lastName` VARCHAR(25) NULL,

`firstName` VARCHAR(25) NULL,

`age` INT NULL,

`gender` CHAR NULL,

`address` VARCHAR(255) NULL,

`deletion\_date` DATETIME NULL,

PRIMARY KEY (`customer\_id`))

ENGINE = InnoDB;

SET SQL\_MODE=@OLD\_SQL\_MODE;

SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;

SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;

**DML:**

-- Insert into Product table

INSERT INTO product(VehicleCode,VehicleType,SumInsured,premium) values(001,'BMW',25000,1800);

INSERT INTO product(VehicleCode,VehicleType,SumInsured,premium) values(002,'Audi',22000,1800);

INSERT INTO product(VehicleCode,VehicleType,SumInsured,premium) values(003,'Ducati',20000,1200);

INSERT INTO product(VehicleCode,VehicleType,SumInsured,premium) values(004,'Honda',25030,1200);

INSERT INTO product(VehicleCode,VehicleType,SumInsured,premium) values(005,'Ford',25000,1800);

INSERT INTO product(VehicleCode,VehicleType,SumInsured,premium) values(006,'Mercedece',25000,1800);

INSERT INTO product(VehicleCode,VehicleType,SumInsured,premium) values(007,'Volkswagon',25000,1800);

INSERT INTO product(VehicleCode,VehicleType,SumInsured,premium) values(008,'Chervolet',25000,1800);

-- Insert into CustomerType Table

INSERT INTO customertype values('P',"Policy Holder");

INSERT INTO customertype values('A',"Agent");

/\*

Author: Nilay Virenderkumar Haryal

Title: Create and Update Customers and assign roles to them

Date Created: 12-12-2018

\*/

DROP PROCEDURE IF EXISTS CreateUpdate\_Customer\_proc;

DELIMITER //

CREATE PROCEDURE CreateUpdate\_Customer\_proc(INOUT Customer\_ID\_v int,

IN lName\_v varchar(25),

IN fName\_v varchar(25),

IN age\_v int,

IN gender\_v char,

IN address\_v varchar(255),

IN Type\_of\_customer\_v char,

IN selected\_action\_v INT)

BEGIN

select count(\*) INTO @recordExist from customer c where c.CustomerID=Customer\_ID\_v;

select count(\*) INTO @typeExist from customerType where IDType=Type\_of\_customer\_v;

IF @typeExist = 1 THEN

IF @recordExist = 0 THEN

INSERT INTO customer(LastName,FirstName,age,gender,address) values(lName\_v,fName\_v,age\_v,gender\_v,address\_v);

select max(c.customerID) INTO Customer\_ID\_v from customer c;

select Customer\_ID\_v as 'Customer created with Customer ID';

IF selected\_action\_v =0 THEN

select count(\*) INTO @Notunique

from customer\_has\_customerType

where customer\_customerId=Customer\_ID\_v

and customerType\_IDType=Type\_of\_customer\_v;

IF @Notunique=0 then

INSERT INTO customer\_has\_customerType values(Customer\_ID\_v,Type\_of\_customer\_v);

else

signal sqlstate '45000' set message\_text="Such role already exists for the given customer ID";

END IF;

ELSEIF selected\_action\_v =1 then

select count(\*) INTO @Notunique

from customer\_has\_customerType

where customer\_customerId=Customer\_ID\_v;

IF @Notunique=0 then

signal sqlstate '45000' set message\_text="No role exists for the given customer ID to update";

else

UPDATE customer\_has\_customerType c

SET customerType\_IDType=Type\_of\_customer\_v

WHERE customer\_customerId=Customer\_ID\_v;

END IF;

END IF;

ELSE

UPDATE customer c SET

c.LastName = lName\_v,

c.FirstName= fName\_v,

c.age= age\_v,

c.gender= gender\_v,

c.address= address\_v

WHERE c.CustomerID=Customer\_ID\_v;

select "Customer Updated" as "Alert";

IF selected\_action\_v =0 THEN

select count(\*) INTO @Notunique

from customer\_has\_customerType

where customer\_customerId=Customer\_ID\_v

and customerType\_IDType=Type\_of\_customer\_v;

IF @Notunique=0 then

INSERT INTO customer\_has\_customerType values(Customer\_ID\_v,Type\_of\_customer\_v);

else

signal sqlstate '45000' set message\_text="Such role already exists for the given customer ID";

END IF;

ELSEIF selected\_action\_v =1 then

select count(\*) INTO @Notunique

from customer\_has\_customerType

where customer\_customerId=Customer\_ID\_v;

IF @Notunique=0 then

signal sqlstate '45000' set message\_text="No role exists for the given customer ID to update";

else

UPDATE customer\_has\_customerType c

SET customerType\_IDType=Type\_of\_customer\_v

WHERE customer\_customerId=Customer\_ID\_v;

END IF;

END IF;

END IF;

ELSE

signal sqlstate '45000' set message\_text="[BR1]Customer can be either Policy Holder(P) or Agent(A)";

END IF;

END;

//

/\*

SET @CustomerID=null;

SET @lName\_v='Haryal';

SET @fName\_v='Nilay' ;

SET @age\_v=27;

SET @gender\_v='M';

SET @address\_v ='115 Saint Stephen Street';

SET @TypeOfCustomer='P';

-- 0 = insert

-- 1 = update

call CreateUpdate\_Customer\_proc(@CustomerID,@lName\_v,@fName\_v,@age\_v,@gender\_v,@address\_v,@TypeOfCustomer,1);

\*/

/\*

Author: Nilay Virenderkumar Haryal

Title: Create quote in the Quotation Table

Date Created: 12-12-2018

\*/

use motor\_insurance;

DROP FUNCTION IF EXISTS createQuote\_fn;

DELIMITER //

CREATE FUNCTION createQuote\_fn(Quotation\_No varchar(25),

Start\_Date date,

End\_Date date,

vehicle\_Code int,

customer\_ID int,

agent\_ID int,

ComPercent double,

Type\_Of\_Quote char(4),

sum\_Insured INT,

prem INT) RETURNS VARCHAR(25) DETERMINISTIC

BEGIN

DECLARE cur1 CURSOR FOR SELECT p.SumInsured,p.Premium FROM Product p where p.VehicleCode=vehicle\_Code;

OPEN cur1;

FETCH cur1 INTO sum\_Insured,prem;

CLOSE cur1;

INSERT INTO Quotation(StartDate,EndDate,VehicleCode,CustomerID,AgentID,ComissionPercent,TypeOfQuote,SumInsured,Premium,timestamp)

values(Start\_Date,End\_Date,vehicle\_Code,customer\_ID,agent\_ID,ComPercent,Type\_Of\_Quote,sum\_Insured,prem,now());

SELECT max(TransactionNo) INTO @Transaction\_no from Quotation;

UPDATE Quotation SET QuoteNo=ifnull(Quotation\_No,concat(left(Type\_Of\_Quote,1),@Transaction\_no)) where TransactionNo=@Transaction\_no;

SET @Quote\_no=ifnull(Quotation\_No,concat(left(Type\_Of\_Quote,1),@Transaction\_no));

RETURN @Quote\_no;

END;

//

/\*

Author: Nilay Virenderkumar Haryal

Title: Create and Update Quote

Date Created: 12-12-2018

\*/

use motor\_insurance;

DROP PROCEDURE IF EXISTS CreateUpdateQuote\_proc;

DELIMITER //

CREATE PROCEDURE CreateUpdateQuote\_proc(INOUT Quotation\_No varchar(25),

IN Start\_Date datetime,

IN End\_Date datetime,

IN vehicle\_Code int,

IN customer\_ID int,

IN agent\_ID int,

IN ComPercent double,

IN TypeOfQuote char(4))

BEGIN

DECLARE sum\_Insured,prem INT;

DECLARE cur1 CURSOR FOR SELECT p.SumInsured,p.Premium FROM Product p where p.VehicleCode=vehicle\_Code;

SELECT count(\*) INTO @vehicleCodeExist FROM Product p where p.VehicleCode=vehicle\_Code;

select count(\*) INTO @customerIDExist FROM customer\_has\_customertype c where customer\_customerId=customer\_ID and customerType\_IDType ='P';

select count(\*) INTO @agentIDExist FROM customer\_has\_customertype c where customer\_customerId=agent\_ID and customerType\_IDType ='A';

select count(\*) INTO @quotationExist FROM Quotation q where q.QuoteNo=Quotation\_No and q.typeOfQuote='Q'and TransactionNo=(select max(qu.TransactionNo) from Quotation qu where qu.QuoteNo=Quotation\_No);

IF @vehicleCodeExist=0 OR @customerIDExist=0 OR @agentIDExist=0 THEN

signal sqlstate '45000' set message\_text="Invalid Vehicle\_Code/CustomerID/AgentID";

ELSE

OPEN Cur1;

FETCH Cur1 INTO sum\_Insured,prem;

CLOSE Cur1;

IF TypeOfQuote='Q' THEN

IF @quotationExist=0 THEN

SELECT createQuote\_fn(null,Start\_Date,End\_Date,vehicle\_Code,customer\_ID,agent\_ID,ComPercent,TypeOfQuote,sum\_Insured,prem)

AS "Quote Created with Quotation Number:";

ELSE

UPDATE Quotation

SET startDate=Start\_Date,

endDate=End\_Date,

vehicleCode=vehicle\_Code,

customerID=customer\_ID,

agentID=agent\_ID,

comissionPercent=ComPercent,

sumInsured=sum\_Insured,

premium= prem

WHERE Quotation\_No=QuoteNo;

select "Quote has been updated" as "Alert";

END IF;

ELSEIF TypeOfQuote='QC' THEN

IF @quotationExist=0 THEN

signal sqlstate '45000' set message\_text="No Quote available for given Quotation No";

ELSE

select QuoteNo,startDate,endDate,vehicleCode,customerID,agentID,comissionPercent,sumInsured,premium

INTO @QuoteNo\_v,@startDate\_v,@endDate\_v,@vehicleCode\_v,@customerID\_v,@agentID\_v,@comissionPercent\_v,@sumInsured\_v,@premium\_v

from Quotation where TransactionNo=(select max(qu.TransactionNo) from Quotation qu where qu.QuoteNo=Quotation\_No);

SELECT createQuote\_fn(@QuoteNo\_v,@startDate\_v,@endDate\_v,@vehicleCode\_v,@customerID\_v,@agentID\_v,@comissionPercent\_v,'QC',@sumInsured\_v,@premium\_v)

AS "Quote Cancelled with Quotation Number:";

END IF;

ELSE

signal sqlstate '45000' set message\_text="Please select a valid value for the field Policy Type";

END IF;

END IF;

END;

//

/\*

set @Quote\_no=null;

set @StartDate='2018-12-21';

set @EndDate='2019-12-21';

set @Vehicle\_code='001';

set @customer\_ID=1;

set @agent\_ID=1;

set @ComPercent=15;

set @TypeOfQuote='Q';

call CreateUpdateQuote\_proc(@Quote\_no,@StartDate,@EndDate,@Vehicle\_code,@customer\_ID,@agent\_ID,@ComPercent,@TypeOfQuote);

\*/

/\*

Author: Nilay Virenderkumar Haryal

Title: Create Policy function

Date Created: 12-12-2018

\*/

use motor\_insurance;

DROP FUNCTION IF EXISTS createPolicy\_fn;

DELIMITER //

CREATE FUNCTION createPolicy\_fn(Quote\_No varchar(25))

RETURNS VARCHAR(25) DETERMINISTIC

BEGIN

select startDate,endDate,vehicleCode,customerID,agentID,comissionPercent,sumInsured,premium

INTO @startDate,@endDate,@vehicleCode,@customerID,@agentID,@comissionPercent,@sumInsured,@premium

from Quotation

where QuoteNo=Quote\_No;

insert into policy(startDate,endDate,vehicleCode,customerID,agentID,comissionPercent,typeOfPolicy,sumInsured,premium,timestamp)

values(@startDate,@endDate,@vehicleCode,@customerID,@agentID,@comissionPercent,'P',@sumInsured,@premium,now());

SELECT max(TransactionNo) INTO @Transaction\_no from Policy;

UPDATE Policy SET PolicyNo=concat('P',@Transaction\_no) where TransactionNo=@Transaction\_no;

SET @Policy\_no=concat('P',@Transaction\_no);

select QuoteNo,startDate,endDate,vehicleCode,customerID,agentID,comissionPercent,sumInsured,premium

INTO @QuoteNo\_v,@startDate\_v,@endDate\_v,@vehicleCode\_v,@customerID\_v,@agentID\_v,@comissionPercent\_v,@sumInsured\_v,@premium\_v

from Quotation where TransactionNo=(select max(qu.TransactionNo) from Quotation qu where qu.QuoteNo=Quote\_No);

SET @Q= createQuote\_fn(@QuoteNo\_v,@startDate\_v,@endDate\_v,@vehicleCode\_v,@customerID\_v,@agentID\_v,@comissionPercent\_v,'QC',@sumInsured\_v,@premium\_v);

RETURN @Policy\_no;

END;

//

/\*

Author: Nilay Virenderkumar Haryal

Title: Convert Quote into Policy or Update/Cancel the Policy

Date Created: 12-12-2018

\*/

use motor\_insurance;

DROP PROCEDURE IF EXISTS Convert\_cancel\_policy\_proc;

DELIMITER //

CREATE PROCEDURE Convert\_cancel\_policy\_proc(INOUT Quote\_No varchar(25),IN TypeOfPolicy char(4))

BEGIN

DECLARE prem,ComPercent INT;

DECLARE cur1 CURSOR FOR SELECT q.comissionPercent,q.Premium FROM Quotation q where q.QuoteNo=Quote\_No;

DECLARE cur2 CURSOR FOR SELECT p.comissionPercent,p.Premium FROM Policy p where p.PolicyNo=Quote\_No;

DECLARE exit handler for sqlexception

BEGIN

select "Exception Occured" as "Alert";

ROLLBACK;

END;

START TRANSACTION;

IF TypeOfPolicy='P' THEN

SELECT COUNT(\*) INTO @QuoteExist FROM Quotation q where q.QuoteNo=Quote\_No and q.TypeOfQuote='Q' and TransactionNo=(select max(qu.TransactionNo) from Quotation qu where qu.QuoteNo=Quote\_No);

IF @QuoteExist=0 THEN

SELECT concat("No Quotation avaiable for the given Quote number: ",Quote\_No) as "Alert";

ELSE

OPEN cur1;

FETCH cur1 INTO ComPercent,prem;

CLOSE cur1;

SET @policy\_no= createPolicy\_fn(Quote\_No);

SELECT @policy\_no AS "Policy Created with Policy Number:";

SET @acc\_item\_no= generateAccItems\_fn(prem,(select max(TransactionNo) from policy where PolicyNo=@policy\_no),'CR');

SET @comm\_item = prem \* (ComPercent/100);

SET @acc\_item\_no= generateAccItems\_fn(@comm\_item,(select max(TransactionNo) from policy where PolicyNo=@policy\_no),'CR');

END IF;

ELSEIF TypeOfPolicy='PC' THEN

SELECT COUNT(\*) INTO @PolicyExist FROM Policy p where p.PolicyNo=Quote\_No and p.typeofPolicy='P';

IF @PolicyExist=0 THEN

SELECT concat("No Quotation avaiable for the given Quote number: ",Quote\_No) as "Alert";

ELSE

OPEN cur2;

FETCH cur2 INTO ComPercent,prem;

CLOSE cur2;

UPDATE policy

SET typeofPolicy='PC'

where policyNo= Quote\_No;

SELECT Quote\_No AS "Policy Cancelled with Policy Number:";

SET @acc\_item\_no= generateAccItems\_fn(prem,(select max(TransactionNo) from policy where PolicyNo=@policy\_no),'DR');

SET @comm\_item = prem \* (ComPercent/100);

SET @acc\_item\_no= generateAccItems\_fn(@comm\_item,(select max(TransactionNo) from policy where PolicyNo=@policy\_no),'DR');

END IF;

else

signal sqlstate '45000' set message\_text="Please select a valid value for the field Policy Type";

END IF;

COMMIT;

END;

//

/\*

SET @Policy\_No='P2';

SET @TypeOfPolicy='PC';

CALL Convert\_cancel\_policy\_proc(@Policy\_No,@TypeOfPolicy);

\*/

/\*

Author: Nilay Virenderkumar Haryal

Title: Generate Accounting Items having Debit and Credit entries for every transaction

Date Created: 12-12-2018

\*/

use motor\_insurance;

DROP FUNCTION IF EXISTS generateAccItems\_fn;

DELIMITER //

CREATE FUNCTION generateAccItems\_fn(amt DOUBLE,transaction\_no int,CreditDebit char(2)) RETURNS INT DETERMINISTIC

BEGIN

INSERT INTO acc\_item(amount,Policy\_TransactionNo,`CR/DR`) values(amt,transaction\_no,CreditDebit);

SELECT max(item\_no) INTO @item\_no FROM acc\_item;

return @item\_no;

END;

//

/\*

Author: Nilay Virenderkumar Haryal

Title: Make Payment against issued Policy

Date Created: 12-12-2018

\*/

use motor\_insurance;

drop procedure if exists make\_payment\_proc;

DELIMITER //

create procedure make\_payment\_proc(in amount\_v double,in policy\_no\_v varchar(25))

BEGIN

select TransactionNo,typeOfPolicy INTO @transNo,@typeOfPolicy

from Policy

where policyNo=policy\_no\_v

and TransactionNo=(SELECT max(transactionNo) from policy where policyNo=policy\_no\_v);

IF @typeOfPolicy='P' then

SET @item=generateAccItems\_fn(amount\_v,@transNo,'DR');

SET @balance=(select sum(amount) from acc\_item where Policy\_TransactionNo=@transNo and `CR/DR`='CR')-

(select sum(amount) from acc\_item where Policy\_TransactionNo=@transNo and `CR/DR`='DR');

IF @balance>0 then

SET @status="PARTIALLY PAID";

INSERT INTO acc\_payment(amount,paid\_status,Policy\_TransactionNo)

values(amount\_v,@status,@transNo);

elseif @balance=0 then

SET @status="PAID";

INSERT INTO acc\_payment(amount,paid\_status,Policy\_TransactionNo)

values(amount\_v,@status,@transNo);

elseif @balance<0 then

SET @status="OVER PAID";

SET @REFUND=@balance\*-1;

SET @item=generateAccItems\_fn(@REFUND,@transNo,'CR');

INSERT INTO acc\_payment(amount,paid\_status,Policy\_TransactionNo)

values(amount\_v,@status,@transNo);

INSERT INTO acc\_payment(amount,paid\_status,Policy\_TransactionNo)

values(@REFUND,"REFUND",@transNo);

end if;

else

signal sqlstate '45000' set message\_text="No such valid policy exists";

end if;

END;

//

/\*

SET @amt=1000;

SET @policy\_no='P3';

call make\_payment\_proc(@amt,@policy\_no);

\*/

/\*

Author: Nilay Virenderkumar Haryal

Title: Create claim and approve it

Date Created: 12-12-2018

\*/

use motor\_insurance;

DROP PROCEDURE IF EXISTS create\_claim\_proc;

DELIMITER //

CREATE PROCEDURE create\_claim\_proc(IN claim\_type\_v VARCHAR(45),

IN claim\_amount\_v DOUBLE,

IN incident\_date\_v DATETIME,

IN Policy\_No\_v VARCHAR(25))

BEGIN

SELECT max(transactionNo) INTO @policy\_trans\_no from policy where policyNo=Policy\_No\_v and typeOfPolicy='P';

IF ifnull(@policy\_trans\_no,0) = 0 then

signal sqlstate '45000' set message\_text="No such policy exists";

else

INSERT INTO Claim\_case(claim\_type,claim\_amount,claim\_date,incident\_date,Policy\_TransactionNo) values(claim\_type\_v,claim\_amount\_v,now(),incident\_date\_v,@policy\_trans\_no);

SET @sum\_insured = (select premium from policy where transactionNo=@policy\_trans\_no);

SELECT paid\_status INTO @paid

from acc\_payment

where Policy\_TransactionNo=

(select cc.Policy\_TransactionNo from claim\_case cc where cc.case\_no=(select max(cq.case\_no) from claim\_case cq))

order by acc\_payment\_no desc limit 1;

IF (2\*@sum\_insured)<claim\_amount\_v OR @paid ="PARTIALLY PAID" then

SET @sum\_approved = 0;

SET @claim\_status= "REJECTED";

else

SET @sum\_approved = if(@sum\_insured<claim\_amount\_v,@sum\_insured,claim\_amount\_v);

SET @claim\_status=if(@sum\_insured<claim\_amount\_v,"PARTIAL APPROVAL","APPROVED");

end if;

INSERT INTO claim(sum\_insured,sum\_requested,sum\_approved,claim\_status,claim\_case\_claim\_no)

values(@sum\_insured,claim\_amount\_v,@sum\_approved,@claim\_status,(select max(case\_no) from claim\_case));

select c.sum\_approved,cc.Policy\_TransactionNo

INTO @amt,@trans\_no

from claim\_case cc,claim c

where c.claim\_case\_claim\_no=cc.case\_no

and claim\_no=(select max(claim\_no) from claim);

INSERT INTO acc\_item(amount,`CR/DR`,Policy\_TransactionNo)

values(@amt,'DR',@trans\_no);

End if;

END;

//

/\*

SET @claim\_type='accident';

SET @claim\_amount\_v='4000';

SET @incident\_date\_v='2018-12-11';

SET @Policy\_No='P3';

call create\_claim\_proc(@claim\_type,@claim\_amount\_v,@incident\_date\_v,@Policy\_No);

\*/

/\*

Author: Nilay Virenderkumar Haryal

Title: Trigger to maintain customer history

Date Created: 12-12-2018

\*/

use motor\_insurance;

DROP Trigger if EXISTS customerHistory\_trigger;

DELIMITER //

CREATE TRIGGER customerHistory

BEFORE DELETE ON customer

FOR EACH ROW

BEGIN

INSERT INTO customer\_history values(old.customerId,old.lastName,old.firstName,old.age,old.gender,old.address,now());

END//

DELIMITER ;

/\*

Author: Nilay Virenderkumar Haryal

Title: Renewal reminder View

Date Created: 12-12-2018

\*/

CREATE OR REPLACE VIEW renewal\_reminder\_vw

AS

select c.customerId "Customer ID",

concat\_ws(' ',c.firstName,c.lastName) "Name",

c.age "Age",

c.gender "Gender",

c.address "Address",

p.policyNo "Policy No",

p.startdate "Policy Start Date",

p.enddate "Policy End Date",

datediff(p.enddate,p.startdate) "Days to expiry"

from customer c,policy p

where c.customerId=p.customerId

and typeofpolicy='P'

order by "Days to expiry" desc;

use motor\_insurance;

CREATE OR REPLACE VIEW payment\_reminder\_vw

AS

select c.customerId "Customer ID",

concat\_ws(' ',c.firstName,c.lastName) "Name",

c.age "Age",

c.gender "Gender",

c.address "Address",

p.policyNo "Policy No",

ap.paid\_status "Payment Status"

from Policy p, acc\_payment ap,Customer c

where c.customerID=p.CustomerID

and acc\_payment\_no=(select max(acp.acc\_payment\_no) from acc\_payment acp where acp.Policy\_TransactionNo=p.TransactionNo)

and ap.paid\_status NOT IN ('REFUND','PAID');

/\*

Author: Nilay Virenderkumar Haryal

Title: Backup

Date Created: 13-12-2018

\*/

CREATE EVENT IF NOT EXISTS backup\_policy

ON SCHEDULE EVERY 1 WEEK

DO

CREATE TABLE policy\_backup as select \* from policy;

/\*

Author: Nilay Virenderkumar Haryal

Title: View for accounting items

Date Created: 13-12-2018

\*/

CREATE OR REPLACE VIEW acc\_items\_vw AS

select sum(ac.amount) "CR amount",

sum(acc.amount) "DR amount",

ac.Policy\_TransactionNo "Policy\_TransactionNo",

p.PolicyNo "PolicyNo",

ap.paid\_status "Payment Status",

p.customerId "Customer ID"

from acc\_item ac,acc\_item acc,policy p,acc\_payment ap,claim\_case cc,claim c

where ac.Policy\_TransactionNo=acc.Policy\_TransactionNo

and ac.Policy\_TransactionNo=p.TransactionNo

and ac.Policy\_TransactionNo=ap.Policy\_TransactionNo

and ac.`CR/DR`='CR'

and acc.`CR/DR`='DR'

group by ac.Policy\_TransactionNo;

/\* Assigning Roles \*/

use motor\_insurance;

create user 'policyholder'@'localhost' IDENTIFIED BY 'p1';

create user 'agent'@'localhost' IDENTIFIED BY 'a1';

create user 'companyagent'@'localhost' IDENTIFIED BY 'c1';

GRANT ALL PRIVILEGES ON procedure motor\_insurance.CreateUpdate\_Customer\_proc TO 'companyagent'@'localhost';

GRANT ALL PRIVILEGES ON customer TO 'companyagent'@'localhost';

GRANT ALL PRIVILEGES ON customer\_has\_customer\_type TO 'companyagent'@'localhost';

GRANT ALL PRIVILEGES ON customertype TO 'companyagent'@'localhost';

GRANT ALL PRIVILEGES ON customer\_history TO 'companyagent'@'localhost';

GRANT ALL PRIVILEGES ON PRODUCT TO 'companyagent'@'localhost';

GRANT INSERT ON acc\_item TO 'companyagent'@'localhost';

GRANT INSERT ON acc\_payment TO 'companyagent'@'localhost';

GRANT INSERT ON acc\_payment\_item TO 'companyagent'@'localhost';

GRANT ALL PRIVILEGES ON PROCEDURE Convert\_cancel\_policy\_proc TO 'agent'@'localhost';

GRANT ALL PRIVILEGES ON PROCEDURE create\_claim\_proc TO 'agent'@'localhost';

create\_claim\_proc

GRANT ALL PRIVILEGES ON Quotation TO 'agent'@'localhost';

GRANT ALL PRIVILEGES ON Policy TO 'agent'@'localhost';

GRANT ALL PRIVILEGES ON claim\_case TO 'agent'@'localhost';

GRANT ALL PRIVILEGES ON claim TO 'agent'@'localhost';

GRANT SELECT ON claim TO 'policyholder'@'localhost';

GRANT SELECT ON acc\_payment TO 'policyholder'@'localhost';

**MySQL dump**

-- MySQL dump 10.13 Distrib 8.0.12, for Win64 (x86\_64)

--

-- Host: localhost Database: motor\_insurance

-- ------------------------------------------------------

-- Server version 8.0.12

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

SET NAMES utf8mb4 ;

/\*!40103 SET @OLD\_TIME\_ZONE=@@TIME\_ZONE \*/;

/\*!40103 SET TIME\_ZONE='+00:00' \*/;

/\*!40014 SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0 \*/;

/\*!40014 SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0 \*/;

/\*!40101 SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='NO\_AUTO\_VALUE\_ON\_ZERO' \*/;

/\*!40111 SET @OLD\_SQL\_NOTES=@@SQL\_NOTES, SQL\_NOTES=0 \*/;

--

-- Table structure for table `acc\_item`

--

DROP TABLE IF EXISTS `acc\_item`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `acc\_item` (

`item\_no` int(11) NOT NULL AUTO\_INCREMENT,

`amount` double DEFAULT NULL,

`CR/DR` char(2) NOT NULL,

`Policy\_TransactionNo` int(11) NOT NULL DEFAULT '0',

PRIMARY KEY (`item\_no`),

KEY `fk\_acc\_item\_Policy1\_idx` (`Policy\_TransactionNo`),

CONSTRAINT `fk\_acc\_item\_Policy1` FOREIGN KEY (`Policy\_TransactionNo`) REFERENCES `policy` (`transactionno`)

) ENGINE=InnoDB AUTO\_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `acc\_item`

--

LOCK TABLES `acc\_item` WRITE;

/\*!40000 ALTER TABLE `acc\_item` DISABLE KEYS \*/;

INSERT INTO `acc\_item` VALUES (1,1800,'CR',3),(2,270,'CR',3),(3,1800,'DR',3),(4,1000,'DR',3),(5,730,'CR',3),(6,1800,'CR',4),(7,270,'CR',4),(8,1000,'DR',4),(9,1000,'DR',4),(10,1000,'DR',4),(11,930,'CR',4),(12,0,'DR',4),(13,1800,'DR',4),(14,1800,'CR',5),(15,270,'CR',5),(16,1000,'DR',5),(17,1000,'DR',5),(18,1000,'DR',5),(19,930,'CR',5),(20,1000,'DR',5);

/\*!40000 ALTER TABLE `acc\_item` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `acc\_payment`

--

DROP TABLE IF EXISTS `acc\_payment`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `acc\_payment` (

`acc\_payment\_no` int(11) NOT NULL AUTO\_INCREMENT,

`amount` double DEFAULT NULL,

`paid\_status` varchar(45) DEFAULT NULL,

`Policy\_TransactionNo` int(11) NOT NULL,

PRIMARY KEY (`acc\_payment\_no`),

KEY `fk\_acc\_payment\_Policy1\_idx` (`Policy\_TransactionNo`),

CONSTRAINT `fk\_acc\_payment\_Policy1` FOREIGN KEY (`Policy\_TransactionNo`) REFERENCES `policy` (`transactionno`)

) ENGINE=InnoDB AUTO\_INCREMENT=11 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `acc\_payment`

--

LOCK TABLES `acc\_payment` WRITE;

/\*!40000 ALTER TABLE `acc\_payment` DISABLE KEYS \*/;

INSERT INTO `acc\_payment` VALUES (1,1000,'OVER PAID',3),(2,730,'REFUND',3),(3,1000,'PARTIALLY PAID',4),(4,1000,'PARTIALLY PAID',4),(5,1000,'OVER PAID',4),(6,930,'REFUND',4),(7,1000,'PARTIALLY PAID',5),(8,1000,'PARTIALLY PAID',5),(9,1000,'OVER PAID',5),(10,930,'REFUND',5);

/\*!40000 ALTER TABLE `acc\_payment` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `claim`

--

DROP TABLE IF EXISTS `claim`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `claim` (

`claim\_no` int(11) NOT NULL AUTO\_INCREMENT,

`sum\_insured` int(11) DEFAULT NULL,

`sum\_requested` int(11) DEFAULT NULL,

`sum\_approved` int(11) DEFAULT NULL,

`claim\_status` varchar(45) DEFAULT NULL,

`Claim\_case\_claim\_no` int(11) NOT NULL,

PRIMARY KEY (`claim\_no`,`Claim\_case\_claim\_no`),

KEY `fk\_claim\_Claim\_case1\_idx` (`Claim\_case\_claim\_no`),

CONSTRAINT `fk\_claim\_Claim\_case1` FOREIGN KEY (`Claim\_case\_claim\_no`) REFERENCES `claim\_case` (`case\_no`)

) ENGINE=InnoDB AUTO\_INCREMENT=8 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `claim`

--

LOCK TABLES `claim` WRITE;

/\*!40000 ALTER TABLE `claim` DISABLE KEYS \*/;

INSERT INTO `claim` VALUES (4,1800,2000,1800,'PARTIAL APPROVAL',4),(5,1800,4000,0,'REJECTED',6),(6,1800,2000,1800,'PARTIAL APPROVAL',7),(7,1800,1000,1000,'APPROVED',8);

/\*!40000 ALTER TABLE `claim` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `claim\_case`

--

DROP TABLE IF EXISTS `claim\_case`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `claim\_case` (

`case\_no` int(11) NOT NULL AUTO\_INCREMENT,

`claim\_type` varchar(45) NOT NULL,

`Claim\_amount` double DEFAULT NULL,

`claim\_date` datetime DEFAULT NULL,

`incident\_date` datetime DEFAULT NULL,

`Policy\_TransactionNo` int(11) NOT NULL,

PRIMARY KEY (`case\_no`),

KEY `fk\_Claim\_Policy1\_idx` (`Policy\_TransactionNo`),

CONSTRAINT `fk\_Claim\_Policy1` FOREIGN KEY (`Policy\_TransactionNo`) REFERENCES `policy` (`transactionno`)

) ENGINE=InnoDB AUTO\_INCREMENT=9 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `claim\_case`

--

LOCK TABLES `claim\_case` WRITE;

/\*!40000 ALTER TABLE `claim\_case` DISABLE KEYS \*/;

INSERT INTO `claim\_case` VALUES (4,'accident',2000,'2018-12-12 18:05:06','2018-12-11 00:00:00',3),(5,'minor accident',4000,'2018-12-12 21:14:59','2018-12-11 00:00:00',4),(6,'minor accident',4000,'2018-12-12 21:15:46','2018-12-11 00:00:00',4),(7,'minor accident',2000,'2018-12-12 21:16:16','2018-12-11 00:00:00',4),(8,'accident',1000,'2018-12-13 04:08:33','2018-12-11 00:00:00',5);

/\*!40000 ALTER TABLE `claim\_case` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `customer`

--

DROP TABLE IF EXISTS `customer`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `customer` (

`customerId` int(11) NOT NULL AUTO\_INCREMENT,

`lastName` varchar(25) NOT NULL,

`firstName` varchar(25) DEFAULT NULL,

`age` int(11) DEFAULT NULL,

`gender` char(1) DEFAULT NULL,

`address` varchar(255) DEFAULT NULL,

PRIMARY KEY (`customerId`)

) ENGINE=InnoDB AUTO\_INCREMENT=9 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `customer`

--

LOCK TABLES `customer` WRITE;

/\*!40000 ALTER TABLE `customer` DISABLE KEYS \*/;

INSERT INTO `customer` VALUES (1,'Haryal','Nilay',27,'M','115 Saint Stephen Street'),(2,'Halder','Ripan',24,'M','115 Saint Stephen Street'),(5,'Haryal','Nilay',27,'M','115 Saint Stephen Street'),(6,'Haryal','Nilay',27,'M','115 Saint Stephen Street'),(7,'Haryal','Nilay',27,'M','115 Saint Stephen Street');

/\*!40000 ALTER TABLE `customer` ENABLE KEYS \*/;

UNLOCK TABLES;

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client = utf8mb4 \*/ ;

/\*!50003 SET character\_set\_results = utf8mb4 \*/ ;

/\*!50003 SET collation\_connection = utf8mb4\_0900\_ai\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode = 'STRICT\_TRANS\_TABLES,NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

/\*!50003 CREATE\*/ /\*!50017 DEFINER=`root`@`localhost`\*/ /\*!50003 TRIGGER `customerHistory` BEFORE DELETE ON `customer` FOR EACH ROW BEGIN

INSERT INTO customer\_history values(old.customerId,old.lastName,old.firstName,old.age,old.gender,old.address,now());

END \*/;;

DELIMITER ;

/\*!50003 SET sql\_mode = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection = @saved\_col\_connection \*/ ;

--

-- Table structure for table `customer\_has\_customertype`

--

DROP TABLE IF EXISTS `customer\_has\_customertype`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `customer\_has\_customertype` (

`customer\_customerId` int(11) NOT NULL,

`customerType\_IDType` char(1) NOT NULL,

PRIMARY KEY (`customer\_customerId`,`customerType\_IDType`),

KEY `fk\_customer\_has\_customerType\_customerType1\_idx` (`customerType\_IDType`),

KEY `fk\_customer\_has\_customerType\_customer1\_idx` (`customer\_customerId`),

CONSTRAINT `fk\_customer\_has\_customerType\_customer1` FOREIGN KEY (`customer\_customerId`) REFERENCES `customer` (`customerid`),

CONSTRAINT `fk\_customer\_has\_customerType\_customerType1` FOREIGN KEY (`customerType\_IDType`) REFERENCES `customertype` (`idtype`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `customer\_has\_customertype`

--

LOCK TABLES `customer\_has\_customertype` WRITE;

/\*!40000 ALTER TABLE `customer\_has\_customertype` DISABLE KEYS \*/;

INSERT INTO `customer\_has\_customertype` VALUES (2,'A'),(1,'P'),(6,'P'),(7,'P');

/\*!40000 ALTER TABLE `customer\_has\_customertype` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `customer\_history`

--

DROP TABLE IF EXISTS `customer\_history`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `customer\_history` (

`customer\_id` int(11) NOT NULL,

`lastName` varchar(25) DEFAULT NULL,

`firstName` varchar(25) DEFAULT NULL,

`age` int(11) DEFAULT NULL,

`gender` char(1) DEFAULT NULL,

`address` varchar(255) DEFAULT NULL,

`deletion\_date` datetime DEFAULT NULL,

PRIMARY KEY (`customer\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `customer\_history`

--

LOCK TABLES `customer\_history` WRITE;

/\*!40000 ALTER TABLE `customer\_history` DISABLE KEYS \*/;

INSERT INTO `customer\_history` VALUES (3,'Yadav','Ananya',23,'F','143 Park Drive','2018-12-12 21:01:50'),(4,'Yadav','Ananya',23,'F','Boston','2018-12-12 21:31:42'),(8,'Halder','Ripan',27,'M','115 Saint Stephen Street','2018-12-13 04:15:57');

/\*!40000 ALTER TABLE `customer\_history` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `customertype`

--

DROP TABLE IF EXISTS `customertype`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `customertype` (

`IDType` char(1) NOT NULL,

`TypeName` varchar(45) DEFAULT NULL,

PRIMARY KEY (`IDType`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `customertype`

--

LOCK TABLES `customertype` WRITE;

/\*!40000 ALTER TABLE `customertype` DISABLE KEYS \*/;

INSERT INTO `customertype` VALUES ('A','Agent'),('P','Policy Holder');

/\*!40000 ALTER TABLE `customertype` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Temporary view structure for view `payment\_reminder\_vw`

--

DROP TABLE IF EXISTS `payment\_reminder\_vw`;

/\*!50001 DROP VIEW IF EXISTS `payment\_reminder\_vw`\*/;

SET @saved\_cs\_client = @@character\_set\_client;

SET character\_set\_client = utf8mb4;

/\*!50001 CREATE VIEW `payment\_reminder\_vw` AS SELECT

1 AS `Customer ID`,

1 AS `Name`,

1 AS `Age`,

1 AS `Gender`,

1 AS `Address`,

1 AS `Policy No`,

1 AS `Payment Status`\*/;

SET character\_set\_client = @saved\_cs\_client;

--

-- Table structure for table `policy`

--

DROP TABLE IF EXISTS `policy`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `policy` (

`policyNo` varchar(25) NOT NULL DEFAULT 'NA',

`startDate` datetime DEFAULT NULL,

`endDate` varchar(45) DEFAULT NULL,

`vehicleCode` int(11) DEFAULT NULL,

`customerID` varchar(45) DEFAULT NULL,

`agentID` varchar(45) DEFAULT NULL,

`comissionPercent` double DEFAULT NULL,

`typeOfPolicy` char(4) DEFAULT NULL,

`sumInsured` double DEFAULT NULL,

`premium` double DEFAULT NULL,

`TransactionNo` int(11) NOT NULL AUTO\_INCREMENT,

`timestamp` datetime DEFAULT NULL,

PRIMARY KEY (`TransactionNo`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `policy`

--

LOCK TABLES `policy` WRITE;

/\*!40000 ALTER TABLE `policy` DISABLE KEYS \*/;

INSERT INTO `policy` VALUES ('P3','2018-12-21 00:00:00','2019-12-21',1,'1','2',15,'P',25000,1800,3,'2018-12-12 17:37:53'),('P4','2018-12-21 00:00:00','2019-12-21 00:00:00',2,'4','2',15,'P',22000,1800,4,'2018-12-12 21:09:31'),('P5','2018-12-21 00:00:00','2019-12-21',1,'8','2',15,'P',25000,1800,5,'2018-12-13 04:02:14');

/\*!40000 ALTER TABLE `policy` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `product`

--

DROP TABLE IF EXISTS `product`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `product` (

`VehicleCode` int(11) NOT NULL,

`VehicleType` varchar(255) NOT NULL,

`SumInsured` double NOT NULL,

`premium` double NOT NULL,

PRIMARY KEY (`VehicleCode`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `product`

--

LOCK TABLES `product` WRITE;

/\*!40000 ALTER TABLE `product` DISABLE KEYS \*/;

INSERT INTO `product` VALUES (1,'BMW',25000,1800),(2,'Audi',22000,1800),(3,'Ducati',20000,1200),(4,'Honda',25030,1200),(5,'Ford',25000,1800),(6,'Mercedece',25000,1800),(7,'Volkswagon',25000,1800),(8,'Chervolet',25000,1800);

/\*!40000 ALTER TABLE `product` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `quotation`

--

DROP TABLE IF EXISTS `quotation`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `quotation` (

`QuoteNo` varchar(25) NOT NULL DEFAULT 'NA',

`startDate` datetime DEFAULT NULL,

`endDate` varchar(45) DEFAULT NULL,

`vehicleCode` int(11) NOT NULL,

`customerID` int(11) DEFAULT NULL,

`agentID` int(11) DEFAULT NULL,

`comissionPercent` double DEFAULT NULL,

`typeOfQuote` char(4) DEFAULT NULL,

`sumInsured` double DEFAULT NULL,

`premium` double DEFAULT NULL,

`TransactionNo` int(11) NOT NULL AUTO\_INCREMENT,

`timestamp` datetime DEFAULT NULL,

`Policy\_TransactionNo` int(11) DEFAULT NULL,

PRIMARY KEY (`TransactionNo`),

KEY `fk\_Policy\_customer0\_idx` (`customerID`),

KEY `fk\_Policy\_customer1\_idx` (`agentID`),

KEY `fk\_Quotation\_Policy1\_idx` (`Policy\_TransactionNo`),

KEY `fk\_Policy\_Product10\_idx` (`vehicleCode`),

CONSTRAINT `fk\_Policy\_Product10` FOREIGN KEY (`vehicleCode`) REFERENCES `product` (`vehiclecode`),

CONSTRAINT `fk\_Policy\_customer0` FOREIGN KEY (`customerID`) REFERENCES `customer` (`customerid`),

CONSTRAINT `fk\_Policy\_customer1` FOREIGN KEY (`agentID`) REFERENCES `customer` (`customerid`),

CONSTRAINT `fk\_Quotation\_Policy1` FOREIGN KEY (`Policy\_TransactionNo`) REFERENCES `policy` (`transactionno`)

) ENGINE=InnoDB AUTO\_INCREMENT=7 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `quotation`

--

LOCK TABLES `quotation` WRITE;

/\*!40000 ALTER TABLE `quotation` DISABLE KEYS \*/;

INSERT INTO `quotation` VALUES ('Q1','2018-12-21 00:00:00','2019-12-21',1,1,2,15,'Q',25000,1800,1,'2018-12-12 17:32:48',NULL),('Q1','2018-12-21 00:00:00','2019-12-21',1,1,2,15,'QC',25000,1800,2,'2018-12-12 17:37:53',NULL);

/\*!40000 ALTER TABLE `quotation` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Temporary view structure for view `renewal\_reminder\_vw`

--

DROP TABLE IF EXISTS `renewal\_reminder\_vw`;

/\*!50001 DROP VIEW IF EXISTS `renewal\_reminder\_vw`\*/;

SET @saved\_cs\_client = @@character\_set\_client;

SET character\_set\_client = utf8mb4;

/\*!50001 CREATE VIEW `renewal\_reminder\_vw` AS SELECT

1 AS `Customer ID`,

1 AS `Name`,

1 AS `Age`,

1 AS `Gender`,

1 AS `Address`,

1 AS `Policy No`,

1 AS `Policy Start Date`,

1 AS `Policy End Date`,

1 AS `Days to expiry`\*/;

SET character\_set\_client = @saved\_cs\_client;

--

-- Dumping routines for database 'motor\_insurance'

--

/\*!50003 DROP FUNCTION IF EXISTS `createPolicy\_fn` \*/;

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client = utf8mb4 \*/ ;

/\*!50003 SET character\_set\_results = utf8mb4 \*/ ;

/\*!50003 SET collation\_connection = utf8mb4\_0900\_ai\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode = 'STRICT\_TRANS\_TABLES,NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` FUNCTION `createPolicy\_fn`(Quote\_No varchar(25)) RETURNS varchar(25) CHARSET utf8mb4

DETERMINISTIC

BEGIN

select startDate,endDate,vehicleCode,customerID,agentID,comissionPercent,sumInsured,premium

INTO @startDate,@endDate,@vehicleCode,@customerID,@agentID,@comissionPercent,@sumInsured,@premium

from Quotation

where QuoteNo=Quote\_No;

insert into policy(startDate,endDate,vehicleCode,customerID,agentID,comissionPercent,typeOfPolicy,sumInsured,premium,timestamp)

values(@startDate,@endDate,@vehicleCode,@customerID,@agentID,@comissionPercent,'P',@sumInsured,@premium,now());

SELECT max(TransactionNo) INTO @Transaction\_no from Policy;

UPDATE Policy SET PolicyNo=concat('P',@Transaction\_no) where TransactionNo=@Transaction\_no;

SET @Policy\_no=concat('P',@Transaction\_no);

select QuoteNo,startDate,endDate,vehicleCode,customerID,agentID,comissionPercent,sumInsured,premium

INTO @QuoteNo\_v,@startDate\_v,@endDate\_v,@vehicleCode\_v,@customerID\_v,@agentID\_v,@comissionPercent\_v,@sumInsured\_v,@premium\_v

from Quotation where TransactionNo=(select max(qu.TransactionNo) from Quotation qu where qu.QuoteNo=Quote\_No);

SET @Q= createQuote\_fn(@QuoteNo\_v,@startDate\_v,@endDate\_v,@vehicleCode\_v,@customerID\_v,@agentID\_v,@comissionPercent\_v,'QC',@sumInsured\_v,@premium\_v);

RETURN @Policy\_no;

END ;;

DELIMITER ;

/\*!50003 SET sql\_mode = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection = @saved\_col\_connection \*/ ;

/\*!50003 DROP FUNCTION IF EXISTS `createQuote\_fn` \*/;

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client = utf8mb4 \*/ ;

/\*!50003 SET character\_set\_results = utf8mb4 \*/ ;

/\*!50003 SET collation\_connection = utf8mb4\_0900\_ai\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode = 'STRICT\_TRANS\_TABLES,NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` FUNCTION `createQuote\_fn`(Quotation\_No varchar(25),

Start\_Date date,

End\_Date date,

vehicle\_Code int,

customer\_ID int,

agent\_ID int,

ComPercent double,

Type\_Of\_Quote char(4),

sum\_Insured INT,

prem INT) RETURNS varchar(25) CHARSET utf8mb4

DETERMINISTIC

BEGIN

DECLARE cur1 CURSOR FOR SELECT p.SumInsured,p.Premium FROM Product p where p.VehicleCode=vehicle\_Code;

OPEN cur1;

FETCH cur1 INTO sum\_Insured,prem;

CLOSE cur1;

INSERT INTO Quotation(StartDate,EndDate,VehicleCode,CustomerID,AgentID,ComissionPercent,TypeOfQuote,SumInsured,Premium,timestamp)

values(Start\_Date,End\_Date,vehicle\_Code,customer\_ID,agent\_ID,ComPercent,Type\_Of\_Quote,sum\_Insured,prem,now());

SELECT max(TransactionNo) INTO @Transaction\_no from Quotation;

UPDATE Quotation SET QuoteNo=ifnull(Quotation\_No,concat(left(Type\_Of\_Quote,1),@Transaction\_no)) where TransactionNo=@Transaction\_no;

SET @Quote\_no=ifnull(Quotation\_No,concat(left(Type\_Of\_Quote,1),@Transaction\_no));

RETURN @Quote\_no;

END ;;

DELIMITER ;

/\*!50003 SET sql\_mode = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection = @saved\_col\_connection \*/ ;

/\*!50003 DROP FUNCTION IF EXISTS `generateAccItems\_fn` \*/;

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client = utf8mb4 \*/ ;

/\*!50003 SET character\_set\_results = utf8mb4 \*/ ;

/\*!50003 SET collation\_connection = utf8mb4\_0900\_ai\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode = 'STRICT\_TRANS\_TABLES,NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` FUNCTION `generateAccItems\_fn`(amt DOUBLE,transaction\_no int,CreditDebit char(2)) RETURNS int(11)

DETERMINISTIC

BEGIN

INSERT INTO acc\_item(amount,Policy\_TransactionNo,`CR/DR`) values(amt,transaction\_no,CreditDebit);

SELECT max(item\_no) INTO @item\_no FROM acc\_item;

return @item\_no;

END ;;

DELIMITER ;

/\*!50003 SET sql\_mode = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection = @saved\_col\_connection \*/ ;

/\*!50003 DROP PROCEDURE IF EXISTS `Convert\_cancel\_policy\_proc` \*/;

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client = utf8mb4 \*/ ;

/\*!50003 SET character\_set\_results = utf8mb4 \*/ ;

/\*!50003 SET collation\_connection = utf8mb4\_0900\_ai\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode = 'STRICT\_TRANS\_TABLES,NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `Convert\_cancel\_policy\_proc`(INOUT Quote\_No varchar(25),IN TypeOfPolicy char(4))

BEGIN

DECLARE prem,ComPercent INT;

DECLARE cur1 CURSOR FOR SELECT q.comissionPercent,q.Premium FROM Quotation q where q.QuoteNo=Quote\_No;

DECLARE cur2 CURSOR FOR SELECT p.comissionPercent,p.Premium FROM Policy p where p.PolicyNo=Quote\_No;

DECLARE exit handler for sqlexception

BEGIN

ROLLBACK;

END;

/\*

DECLARE exit handler for sqlwarning

BEGIN

ROLLBACK;

END;

\*/

START TRANSACTION;

IF TypeOfPolicy='P' THEN

SELECT COUNT(\*) INTO @QuoteExist FROM Quotation q where q.QuoteNo=Quote\_No and q.TypeOfQuote='Q' and TransactionNo=(select max(qu.TransactionNo) from Quotation qu where qu.QuoteNo=Quote\_No);

IF @QuoteExist=0 THEN

SELECT concat("No Quotation avaiable for the given Quote number: ",Quote\_No) as "Alert";

ELSE

OPEN cur1;

FETCH cur1 INTO ComPercent,prem;

CLOSE cur1;

SET @policy\_no= createPolicy\_fn(Quote\_No);

SELECT @policy\_no AS "Policy Created with Policy Number:";

SET @acc\_item\_no= generateAccItems\_fn(prem,(select max(TransactionNo) from policy where PolicyNo=@policy\_no),'CR');

SET @comm\_item = prem \* (ComPercent/100);

SET @acc\_item\_no= generateAccItems\_fn(@comm\_item,(select max(TransactionNo) from policy where PolicyNo=@policy\_no),'CR');

END IF;

ELSEIF TypeOfPolicy='PC' THEN

SELECT COUNT(\*) INTO @PolicyExist FROM Policy p where p.PolicyNo=Quote\_No and p.typeofPolicy='P';

IF @PolicyExist=0 THEN

SELECT concat("No Quotation avaiable for the given Quote number: ",Quote\_No) as "Alert";

ELSE

OPEN cur2;

FETCH cur2 INTO ComPercent,prem;

CLOSE cur2;

UPDATE policy

SET typeofPolicy='PC'

where policyNo= Quote\_No;

SELECT Quote\_No AS "Policy Cancelled with Policy Number:";

SET @acc\_item\_no= generateAccItems\_fn(prem,(select max(TransactionNo) from policy where PolicyNo=@policy\_no),'DR');

SET @comm\_item = prem \* (ComPercent/100);

SET @acc\_item\_no= generateAccItems\_fn(@comm\_item,(select max(TransactionNo) from policy where PolicyNo=@policy\_no),'DR');

END IF;

else

SELECT "Please select a valid value for the field Policy Type" as "Alert";

END IF;

COMMIT;

END ;;

DELIMITER ;

/\*!50003 SET sql\_mode = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection = @saved\_col\_connection \*/ ;

/\*!50003 DROP PROCEDURE IF EXISTS `CreateUpdateQuote\_proc` \*/;

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client = utf8mb4 \*/ ;

/\*!50003 SET character\_set\_results = utf8mb4 \*/ ;

/\*!50003 SET collation\_connection = utf8mb4\_0900\_ai\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode = 'STRICT\_TRANS\_TABLES,NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `CreateUpdateQuote\_proc`(INOUT Quotation\_No varchar(25),

IN Start\_Date datetime,

IN End\_Date datetime,

IN vehicle\_Code int,

IN customer\_ID int,

IN agent\_ID int,

IN ComPercent double,

IN TypeOfQuote char(4))

BEGIN

DECLARE sum\_Insured,prem INT;

DECLARE cur1 CURSOR FOR SELECT p.SumInsured,p.Premium FROM Product p where p.VehicleCode=vehicle\_Code;

SELECT count(\*) INTO @vehicleCodeExist FROM Product p where p.VehicleCode=vehicle\_Code;

select count(\*) INTO @customerIDExist FROM customer\_has\_customertype c where customer\_customerId=customer\_ID and customerType\_IDType ='P';

select count(\*) INTO @agentIDExist FROM customer\_has\_customertype c where customer\_customerId=agent\_ID and customerType\_IDType ='A';

select count(\*) INTO @quotationExist FROM Quotation q where q.QuoteNo=Quotation\_No and q.typeOfQuote='Q'and TransactionNo=(select max(qu.TransactionNo) from Quotation qu where qu.QuoteNo=Quotation\_No);

IF @vehicleCodeExist=0 OR @customerIDExist=0 OR @agentIDExist=0 THEN

signal sqlstate '45000' set message\_text="Invalid Vehicle\_Code/CustomerID/AgentID";

ELSE

OPEN Cur1;

FETCH Cur1 INTO sum\_Insured,prem;

CLOSE Cur1;

IF TypeOfQuote='Q' THEN

IF @quotationExist=0 THEN

SELECT createQuote\_fn(null,Start\_Date,End\_Date,vehicle\_Code,customer\_ID,agent\_ID,ComPercent,TypeOfQuote,sum\_Insured,prem)

AS "Quote Created with Quotation Number:";

ELSE

UPDATE Quotation

SET startDate=Start\_Date,

endDate=End\_Date,

vehicleCode=vehicle\_Code,

customerID=customer\_ID,

agentID=agent\_ID,

comissionPercent=ComPercent,

sumInsured=sum\_Insured,

premium= prem

WHERE Quotation\_No=QuoteNo;

select "Quote has been updated" as "Alert";

END IF;

ELSEIF TypeOfQuote='QC' THEN

IF @quotationExist=0 THEN

signal sqlstate '45000' set message\_text="No Quote available for given Quotation No";

ELSE

select QuoteNo,startDate,endDate,vehicleCode,customerID,agentID,comissionPercent,sumInsured,premium

INTO @QuoteNo\_v,@startDate\_v,@endDate\_v,@vehicleCode\_v,@customerID\_v,@agentID\_v,@comissionPercent\_v,@sumInsured\_v,@premium\_v

from Quotation where TransactionNo=(select max(qu.TransactionNo) from Quotation qu where qu.QuoteNo=Quotation\_No);

SELECT createQuote\_fn(@QuoteNo\_v,@startDate\_v,@endDate\_v,@vehicleCode\_v,@customerID\_v,@agentID\_v,@comissionPercent\_v,'QC',@sumInsured\_v,@premium\_v)

AS "Quote Cancelled with Quotation Number:";

END IF;

ELSE

signal sqlstate '45000' set message\_text="Please select a valid value for the field Policy Type";

END IF;

END IF;

END ;;

DELIMITER ;

/\*!50003 SET sql\_mode = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection = @saved\_col\_connection \*/ ;

/\*!50003 DROP PROCEDURE IF EXISTS `CreateUpdate\_Customer\_proc` \*/;

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client = utf8mb4 \*/ ;

/\*!50003 SET character\_set\_results = utf8mb4 \*/ ;

/\*!50003 SET collation\_connection = utf8mb4\_0900\_ai\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode = 'STRICT\_TRANS\_TABLES,NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `CreateUpdate\_Customer\_proc`(INOUT Customer\_ID\_v int,

IN lName\_v varchar(25),

IN fName\_v varchar(25),

IN age\_v int,

IN gender\_v char,

IN address\_v varchar(255),

IN Type\_of\_customer\_v char,

IN selected\_action\_v INT)

BEGIN

select count(\*) INTO @recordExist from customer c where c.CustomerID=Customer\_ID\_v;

select count(\*) INTO @typeExist from customerType where IDType=Type\_of\_customer\_v;

IF @typeExist = 1 THEN

IF @recordExist = 0 THEN

INSERT INTO customer(LastName,FirstName,age,gender,address) values(lName\_v,fName\_v,age\_v,gender\_v,address\_v);

select max(c.customerID) INTO Customer\_ID\_v from customer c;

select Customer\_ID\_v as 'Customer created with Customer ID';

IF selected\_action\_v =0 THEN

select count(\*) INTO @Notunique

from customer\_has\_customerType

where customer\_customerId=Customer\_ID\_v

and customerType\_IDType=Type\_of\_customer\_v;

IF @Notunique=0 then

INSERT INTO customer\_has\_customerType values(Customer\_ID\_v,Type\_of\_customer\_v);

else

signal sqlstate '45000' set message\_text="Such role already exists for the given customer ID";

END IF;

ELSEIF selected\_action\_v =1 then

select count(\*) INTO @Notunique

from customer\_has\_customerType

where customer\_customerId=Customer\_ID\_v;

IF @Notunique=0 then

signal sqlstate '45000' set message\_text="No role exists for the given customer ID to update";

else

UPDATE customer\_has\_customerType c

SET customerType\_IDType=Type\_of\_customer\_v

WHERE customer\_customerId=Customer\_ID\_v;

END IF;

END IF;

ELSE

UPDATE customer c SET

c.LastName = lName\_v,

c.FirstName= fName\_v,

c.age= age\_v,

c.gender= gender\_v,

c.address= address\_v

WHERE c.CustomerID=Customer\_ID\_v;

select "Customer Updated" as "Alert";

IF selected\_action\_v =0 THEN

select count(\*) INTO @Notunique

from customer\_has\_customerType

where customer\_customerId=Customer\_ID\_v

and customerType\_IDType=Type\_of\_customer\_v;

IF @Notunique=0 then

INSERT INTO customer\_has\_customerType values(Customer\_ID\_v,Type\_of\_customer\_v);

else

signal sqlstate '45000' set message\_text="Such role already exists for the given customer ID";

END IF;

ELSEIF selected\_action\_v =1 then

select count(\*) INTO @Notunique

from customer\_has\_customerType

where customer\_customerId=Customer\_ID\_v;

IF @Notunique=0 then

signal sqlstate '45000' set message\_text="No role exists for the given customer ID to update";

else

UPDATE customer\_has\_customerType c

SET customerType\_IDType=Type\_of\_customer\_v

WHERE customer\_customerId=Customer\_ID\_v;

END IF;

END IF;

END IF;

ELSE

signal sqlstate '45000' set message\_text="[BR1]Customer can be either Policy Holder(P) or Agent(A)";

END IF;

END ;;

DELIMITER ;

/\*!50003 SET sql\_mode = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection = @saved\_col\_connection \*/ ;

/\*!50003 DROP PROCEDURE IF EXISTS `create\_claim\_proc` \*/;

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client = utf8mb4 \*/ ;

/\*!50003 SET character\_set\_results = utf8mb4 \*/ ;

/\*!50003 SET collation\_connection = utf8mb4\_0900\_ai\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode = 'STRICT\_TRANS\_TABLES,NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `create\_claim\_proc`(IN claim\_type\_v VARCHAR(45),

IN claim\_amount\_v DOUBLE,

IN incident\_date\_v DATETIME,

IN Policy\_No\_v VARCHAR(25))

BEGIN

SELECT max(transactionNo) INTO @policy\_trans\_no from policy where policyNo=Policy\_No\_v and typeOfPolicy='P';

IF ifnull(@policy\_trans\_no,0) = 0 then

select "No such policy exists" as "Alert";

else

INSERT INTO Claim\_case(claim\_type,claim\_amount,claim\_date,incident\_date,Policy\_TransactionNo) values(claim\_type\_v,claim\_amount\_v,now(),incident\_date\_v,@policy\_trans\_no);

SET @sum\_insured = (select premium from policy where transactionNo=@policy\_trans\_no);

SELECT paid\_status INTO @paid

from acc\_payment

where Policy\_TransactionNo=

(select cc.Policy\_TransactionNo from claim\_case cc where cc.case\_no=(select max(cq.case\_no) from claim\_case cq))

order by acc\_payment\_no desc limit 1;

IF (2\*@sum\_insured)<claim\_amount\_v OR @paid ="PARTIALLY PAID" then

SET @sum\_approved = 0;

SET @claim\_status= "REJECTED";

else

SET @sum\_approved = if(@sum\_insured<claim\_amount\_v,@sum\_insured,claim\_amount\_v);

SET @claim\_status=if(@sum\_insured<claim\_amount\_v,"PARTIAL APPROVAL","APPROVED");

end if;

INSERT INTO claim(sum\_insured,sum\_requested,sum\_approved,claim\_status,claim\_case\_claim\_no)

values(@sum\_insured,claim\_amount\_v,@sum\_approved,@claim\_status,(select max(case\_no) from claim\_case));

select c.sum\_approved,cc.Policy\_TransactionNo

INTO @amt,@trans\_no

from claim\_case cc,claim c

where c.claim\_case\_claim\_no=cc.case\_no

and claim\_no=(select max(claim\_no) from claim);

INSERT INTO acc\_item(amount,`CR/DR`,Policy\_TransactionNo)

values(@amt,'DR',@trans\_no);

End if;

END ;;

DELIMITER ;

/\*!50003 SET sql\_mode = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection = @saved\_col\_connection \*/ ;

/\*!50003 DROP PROCEDURE IF EXISTS `make\_payment\_proc` \*/;

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client = utf8mb4 \*/ ;

/\*!50003 SET character\_set\_results = utf8mb4 \*/ ;

/\*!50003 SET collation\_connection = utf8mb4\_0900\_ai\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode = 'STRICT\_TRANS\_TABLES,NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `make\_payment\_proc`(in amount\_v double,in policy\_no\_v varchar(25))

BEGIN

select TransactionNo,typeOfPolicy INTO @transNo,@typeOfPolicy

from Policy

where policyNo=policy\_no\_v

and TransactionNo=(SELECT max(transactionNo) from policy where policyNo=policy\_no\_v);

IF @typeOfPolicy='P' then

SET @item=generateAccItems\_fn(amount\_v,@transNo,'DR');

SET @balance=(select sum(amount) from acc\_item where Policy\_TransactionNo=@transNo and `CR/DR`='CR')-

(select sum(amount) from acc\_item where Policy\_TransactionNo=@transNo and `CR/DR`='DR');

IF @balance>0 then

SET @status="PARTIALLY PAID";

INSERT INTO acc\_payment(amount,paid\_status,Policy\_TransactionNo)

values(amount\_v,@status,@transNo);

elseif @balance=0 then

SET @status="PAID";

INSERT INTO acc\_payment(amount,paid\_status,Policy\_TransactionNo)

values(amount\_v,@status,@transNo);

elseif @balance<0 then

SET @status="OVER PAID";

SET @REFUND=@balance\*-1;

SET @item=generateAccItems\_fn(@REFUND,@transNo,'CR');

INSERT INTO acc\_payment(amount,paid\_status,Policy\_TransactionNo)

values(amount\_v,@status,@transNo);

INSERT INTO acc\_payment(amount,paid\_status,Policy\_TransactionNo)

values(@REFUND,"REFUND",@transNo);

end if;

else

select "No such valid policy exists" as "Alert";

end if;

END ;;

DELIMITER ;

/\*!50003 SET sql\_mode = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection = @saved\_col\_connection \*/ ;

--

-- Final view structure for view `payment\_reminder\_vw`

--

/\*!50001 DROP VIEW IF EXISTS `payment\_reminder\_vw`\*/;

/\*!50001 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!50001 SET @saved\_cs\_results = @@character\_set\_results \*/;

/\*!50001 SET @saved\_col\_connection = @@collation\_connection \*/;

/\*!50001 SET character\_set\_client = utf8mb4 \*/;

/\*!50001 SET character\_set\_results = utf8mb4 \*/;

/\*!50001 SET collation\_connection = utf8mb4\_0900\_ai\_ci \*/;

/\*!50001 CREATE ALGORITHM=UNDEFINED \*/

/\*!50013 DEFINER=`root`@`localhost` SQL SECURITY DEFINER \*/

/\*!50001 VIEW `payment\_reminder\_vw` AS select `c`.`customerId` AS `Customer ID`,concat\_ws(' ',`c`.`firstName`,`c`.`lastName`) AS `Name`,`c`.`age` AS `Age`,`c`.`gender` AS `Gender`,`c`.`address` AS `Address`,`p`.`policyNo` AS `Policy No`,`ap`.`paid\_status` AS `Payment Status` from ((`policy` `p` join `acc\_payment` `ap`) join `customer` `c`) where ((`c`.`customerId` = `p`.`customerID`) and (`ap`.`acc\_payment\_no` = (select max(`acp`.`acc\_payment\_no`) from `acc\_payment` `acp` where (`acp`.`Policy\_TransactionNo` = `p`.`TransactionNo`))) and (`ap`.`paid\_status` not in ('REFUND','PAID'))) \*/;

/\*!50001 SET character\_set\_client = @saved\_cs\_client \*/;

/\*!50001 SET character\_set\_results = @saved\_cs\_results \*/;

/\*!50001 SET collation\_connection = @saved\_col\_connection \*/;

--

-- Final view structure for view `renewal\_reminder\_vw`

--

/\*!50001 DROP VIEW IF EXISTS `renewal\_reminder\_vw`\*/;

/\*!50001 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!50001 SET @saved\_cs\_results = @@character\_set\_results \*/;

/\*!50001 SET @saved\_col\_connection = @@collation\_connection \*/;

/\*!50001 SET character\_set\_client = utf8mb4 \*/;

/\*!50001 SET character\_set\_results = utf8mb4 \*/;

/\*!50001 SET collation\_connection = utf8mb4\_0900\_ai\_ci \*/;

/\*!50001 CREATE ALGORITHM=UNDEFINED \*/

/\*!50013 DEFINER=`root`@`localhost` SQL SECURITY DEFINER \*/

/\*!50001 VIEW `renewal\_reminder\_vw` AS select `c`.`customerId` AS `Customer ID`,concat\_ws(' ',`c`.`firstName`,`c`.`lastName`) AS `Name`,`c`.`age` AS `Age`,`c`.`gender` AS `Gender`,`c`.`address` AS `Address`,`p`.`policyNo` AS `Policy No`,`p`.`startDate` AS `Policy Start Date`,`p`.`endDate` AS `Policy End Date`,(to\_days(`p`.`endDate`) - to\_days(`p`.`startDate`)) AS `Days to expiry` from (`customer` `c` join `policy` `p`) where ((`c`.`customerId` = `p`.`customerID`) and (`p`.`typeOfPolicy` = 'P')) order by 'Days to expiry' desc \*/;

/\*!50001 SET character\_set\_client = @saved\_cs\_client \*/;

/\*!50001 SET character\_set\_results = @saved\_cs\_results \*/;

/\*!50001 SET collation\_connection = @saved\_col\_connection \*/;

/\*!40103 SET TIME\_ZONE=@OLD\_TIME\_ZONE \*/;

/\*!40101 SET SQL\_MODE=@OLD\_SQL\_MODE \*/;

/\*!40014 SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS \*/;

/\*!40014 SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS \*/;

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

/\*!40111 SET SQL\_NOTES=@OLD\_SQL\_NOTES \*/;

-- Dump completed on 2018-12-13 6:14:47