Project

Proceures :

call GetShowroomByLocation('India');

call GetOrderedProducts();

call ShowroomsVehicleCompaniesLocation();

call EmployeeInDepartment(1);

—call CountOrderByDeliveryDate('2015-04-23',@total);

select @total;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CREATE PROCEDURE `EmployeeInDepartment` (departmentid int)

BEGIN

Select employee.employeeID,employee.employeeName,employee.salary,

deptt.DepartmentID,deptt.DepartmentName from Employee employee inner join

Department deptt on employee.departmentID=deptt.DepartmentID where employee.departmentID=departmentid;

END

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

delimiter //

CREATE PROCEDURE GetShowroomByLocation(IN locationname varchar(20))

BEGIN

SELECT location.locationname, showroom.showroomname

FROM showroom inner join location on showroom.LocationID=location.LocationID

WHERE location.locationname =locationname;

END//

delimiter ;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DELIMITER $$

CREATE PROCEDURE CountOrderByDeliveryDate(

IN delivdate VARCHAR(25),

OUT total INT)

BEGIN

SELECT count(orderID)

INTO total

FROM CarCoopers.Order

WHERE deliverydate = delivdate;

END$$

DELIMITER ;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CREATE PROCEDURE `GetOrderedProducts`()

BEGIN

Select product.productID,product.productname,op.orderID,op.CustomerID from

Product product inner join OrderedProduct op on product.productID=op.productID;

END

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CREATE PROCEDURE `ShowroomsVehicleCompaniesLocation` ()

BEGIN

select location.locationid,location.locationname,location.zipcode,vc.companyId, vc.companyname,

showroom.showroomid,showroom.showroomname from VehicleCompany vc inner join Location location

on vc.locationid=location.locationid

inner join Showroom showroom on location.locationid=showroom.locationid;

END

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

delimiter $$

Create procedure `CountNulls`()

BEGIN

SELECT 'customiseVehicleID', COUNT(\*)-count(customiseVehicleID) from CarCoopers.Order

union all

SELECT 'vehicleEnterpriseID', COUNT(\*)-count(vehicleEnterpriseID) from CarCoopers.Order;

End$$

Delimiter ;

\_\_\_\_\_\_\_

DELIMITER $$

CREATE FUNCTION CustomerLevel(p\_creditLimit double) RETURNS VARCHAR(10)

DETERMINISTIC

BEGIN

DECLARE lvl varchar(10);

IF p\_creditLimit > 50000 THEN

SET lvl = 'PLATINUM';

ELSEIF (p\_creditLimit <= 50000 AND p\_creditLimit >= 20000) THEN

SET lvl = 'GOLD';

ELSEIF p\_creditLimit < 20000 THEN

SET lvl = 'SILVER';

END IF;

RETURN (lvl);

END $$

DELIMITER ;

drop function if exists CustomerLevel;

select CustomerLevel(20000) as Level ;

select CustomerName, CustomerLevel(CreditLimit) from customer;

Delimiter $$

Create Procedure backup\_customer\_details()

Begin

CREATE TABLE Customer\_new AS SELECT \* FROM Customer;

CREATE TABLE OnlineCustomer\_new AS SELECT \* FROM OnlineCustomer;

CREATE TABLE ShowroomCustomer\_new AS SELECT \* FROM ShowroomCustomer;

END $$

DELIMITER ;

call backup\_customer\_details();

select \* from Customer\_new;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Trigger:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_

fine:CREATE DEFINER = CURRENT\_USER TRIGGER `CarCoopers`.`Order\_BEFORE\_UPDATE` BEFORE UPDATE ON `Order` FOR EACH ROW

BEGIN

UPDATE Delivery d

set d.deliverydate=new.deliverydate

where d.orderid=new.orderid;

END

\_\_\_\_\_\_\_\_\_\_\_\_

Fine :CREATE DEFINER = CURRENT\_USER TRIGGER `CarCoopers`.`account\_create\_date` BEFORE INSERT ON `UserAccount` FOR EACH ROW

BEGIN

SET NEW.AccountCreationDate=NOW();

END

\_\_\_\_\_\_\_\_\_\_\_

fine :CREATE DEFINER = CURRENT\_USER TRIGGER `Customer\_deleted\_audit` AFTER DELETE ON `Customer` FOR EACH ROW

BEGIN

DECLARE vUser varchar(50);

-- Find username of person performing the DELETE into table

Select USER() into vUser;

-- Insert record into audit table

Insert into CustomerAudit(CustomerID,DeletedDate,DeletedBy) values(old.customerID,NOW(),vUser);

END

\_\_\_\_

fine :CREATE DEFINER = CURRENT\_USER TRIGGER `CarCoopers`.`before\_product\_update` BEFORE UPDATE ON `Product` FOR EACH ROW

BEGIN

INSERT INTO PriceLogs(ProductID,Price) values(old.productid,old.price);

END

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CREATE USER 'readuser'@'localhost' IDENTIFIED BY 'read';

GRANT Select ON \* . \* TO 'readuser'@'localhost';

CREATE USER 'updateuser'@'localhost' IDENTIFIED BY 'update';

GRANT update ON \* . \* TO 'updateuser'@'localhost';

CREATE USER 'definitionuser'@'localhost' IDENTIFIED BY 'definition';

GRANT drop,create ON \* . \* TO 'definitionuser'@'localhost';

CREATE USER 'deleteuser'@'localhost' IDENTIFIED BY 'delete';

GRANT delete ON \* . \* TO 'deleteuser'@'localhost';

select \* from mysql.user;

MYSQL DUMP to dump database commands :

>>Cd /usr/local/mysql-5.7----

* Ls
* bin
* ./mysql –u root –p (connect to the client)
* show databases;
* exit;
* ./mysqldump -u –p CarCoopers > /mysql-out/backup-dec8.sql
* cat /mysql-out/backup-dec8.sql

SELECT customerId,customername INTO OUTFILE '/mysql-out/DumpaTable2.csv'

FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'

LINES TERMINATED BY '\n'

FROM customer;

SELECT \* INTO OUTFILE '/tmp/result.shruti'

FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'

LINES TERMINATED BY '\n'

FROM customer;