1. List the customers. For each customer, indicate which category he or she fall into, and his or her contact information. If you have more than one independent categorization of customers, please indicate which category the customer falls into for all of the categorizations.

(SELECT firstName, lastName, 'NA' AS CompanyName, 'PremierCustomer' AS CustomerType, 'Home' as AddressType, street, city, state, zipcode

FROM premierCustomer

INNER JOIN customer ON premierCustomer.customerID = customer.customerID

INNER JOIN individualCustomer ON premierCustomer.customerID = individualCustomer.customerID)

UNION

(SELECT 'NA', 'NA', coName, 'PremierCustomer', type, street, city, state, zipcode

FROM premierCustomer

INNER JOIN customer ON premierCustomer.customerID = customer.customerID

INNER JOIN corporateCustomer ON premierCustomer.customerID = corporateCustomer.customerID

INNER JOIN addressLine ON customer.customerID = addressLine.customerID)

UNION

(SELECT firstName, lastName,'NA', 'SteadyCustomer','Home', street, city, state, zipcode

FROM steadyCustomer

INNER JOIN customer ON steadyCustomer.customerID = customer.customerID

INNER JOIN individualCustomer ON steadyCustomer.customerID = individualCustomer.customerID)

UNION

(SELECT 'NA', 'NA',coName, 'SteadyCustomer',type, street, city, state, zipcode

FROM steadyCustomer

INNER JOIN customer ON steadyCustomer.customerID = customer.customerID

INNER JOIN corporateCustomer ON steadyCustomer.customerID = corporateCustomer.customerID

INNER JOIN addressLine ON customer.customerID = addressLine.customerID)

UNION

(SELECT firstName, lastName,'NA', 'ProspectiveCustomer', 'Home', street, city, state, zipcode

FROM prospectiveCustomer

INNER JOIN customer ON prospectiveCustomer.customerID = customer.customerID

INNER JOIN individualCustomer ON prospectiveCustomer.customerID = individualCustomer.customerID)

UNION

(SELECT 'NA', 'NA',coName, 'ProspectiveCustomer', type, street, city, state, zipcode

FROM prospectiveCustomer

INNER JOIN customer ON prospectiveCustomer.customerID = customer.customerID

INNER JOIN corporateCustomer ON prospectiveCustomer.customerID = corporateCustomer.customerID

INNER JOIN addressLine ON customer.customerID = addressLine.customerID)

2. For each service visit, list the total cost to the customer for that visit.

SELECT maintenanceVisit.visitID,vin,visitDate,visitTime,serviceTechnicianID,

SUM(maintenanceAction.actionCost)

FROM maintenanceVisit

INNER JOIN visitService ON maintenanceVisit.visitID = visitService.visitID

INNER JOIN maintenanceAction ON visitService.actionID=maintenanceAction.actionID

GROUP BY maintenanceVisit.visitID;

3. List the top three customers in terms of their net spending for the past two years, and the total  
that they have spent in that period.

SELECT \* FROM customer INNER JOIN

(SELECT customer.customerID, SUM(actionCost) AS totalCost2Years

FROM customer

INNER JOIN vehicle ON customer.customerID = vehicle.customerID

INNER JOIN maintenanceVisit ON vehicle.vin = maintenanceVisit.vin

INNER JOIN visitService ON maintenanceVisit.visitID = visitService.visitID

INNER JOIN maintenanceAction ON visitService.actionID = maintenanceAction.actionID

WHERE year(maintenanceVisit.visitDate)>2015

GROUP by customer.customerID) AS t

ON customer.customerID = t.customerID;

--Good but perhaps should we show names instead  
4. Find all of the mechanics who have three or more skills.

SELECT e.lastName, e.firstName, COUNT(skillName) AS count FROM Mechanic m

INNER JOIN Employee e ON m.employeeID = e.employeeID

INNER JOIN MechanicSkills s ON m.employeeID = s.employeeID

GROUP BY s.employeeId

HAVING COUNT >= 3;

5. Find all of the mechanics who have three or more skills in common.  
 Please give the name of each of the two mechanics sharing 3 or more skills.

Please make sure that any given pair of mechanics only shows up once.

SELECT distinct A.LAStName AS 'Mechanic firstName' , A.firstName AS 'Mechanic lastName', B.lastName AS 'Mechanic 2 firstName', B.firstName AS 'Mechanic 2 lastName'

FROM Employee A

INNER JOIN Employee B WHERE A.employeeID IN

(SELECT employeeID FROM

(SELECT Employee.employeeID,

COUNT(skillName) AS totalSkills

FROM Employee

INNER JOIN Mechanic ON Employee.employeeID = Mechanic.employeeID

INNER JOIN MechanicSkills ON Mechanic.employeeID = MechanicSkills.employeeID

GROUP BY employeeID) AS t)

AND A.employeeID > B.employeeID;

6. For each maintenance package, list the total cost of the maintenance package, as well as a list of all of the maintenance items within that package.

(SELECT maintenancePackage.packageName as Package,'Entire Package' as Item, maintenanceAction.actionCost as Cost$

FROM maintenancePackage

INNER JOIN maintenanceAction ON maintenancePackage.actionID = maintenanceAction.actionID)

UNION

(SELECT maintenancePackage.packageName,maintenanceItem.serviceName, maintenanceAction.actionCost

FROM maintenancePackage

INNER JOIN packageItem ON maintenancePackage.actionID = packageItem.packageID

INNER JOIN maintenanceItem ON packageItem.itemID = maintenanceItem.actionID

INNER JOIN maintenanceAction ON maintenanceItem.actionID = maintenanceAction.actionID);

7. Find all of those mechanics who have one or more maintenance items that they lacked one or  
more of the necessary skills.

select distinct MechanicTask.employeeID ,

MechanicTask.skillName,

itemSkills.skillName

from MechanicTask

inner join

itemSkills

on MechanicTask.actionID = itemSkills.actionID

where MechanicTask.skillName <> itemSkills.skillName;

--Incomplete

8. List the customers, sorted by the number of loyalty points that they have, from largest to  
smallest.

(SELECT lastName, firstName, 'NA' AS coName, loyaltyPoints

FROM steadyCustomer

INNER JOIN individualCustomer ON steadyCustomer.customerID = individualCustomer.customerID

ORDER BY loyaltyPoints DESC)

UNION

(SELECT 'NA', 'NA', coName, loyaltyPoints

FROM steadyCustomer

INNER JOIN corporateCustomer ON steadyCustomer.customerID = corporateCustomer.customerID

ORDER BY loyaltyPoints DESC);

9. List the premier customers and the difference between what they have paid in the past year,

versus the services that they actually used during that same time. List from the customers with  
the largest difference to the smallest.

SELECT c.customerID, annualFee - sum(actionCost) AS Difference FROM premierCustomer c INNER JOIN vehicle v ON c.customerID = v.customerID

INNER JOIN maintenanceVisit m ON v.vin = m.vin

INNER JOIN visitService s ON m.visitID = s.visitID

INNER JOIN maintenanceAction i ON s.actionID = i.actionID

group by customerID

--Unknown column 'annualFee' in 'field list’

10. Report on the steady customers based on the net profit that we have made from them over the  
past year, and the dollar amount of that profit, in order from the greatest to the least. (OLD)

SELECT c.customerID, sum(actionCost) AS netProfit FROM steadyCustomer c INNER JOIN vehicle v ON c.customerID = v.customerID

INNER JOIN maintenanceVisit m ON v.vin = m.vin

INNER JOIN visitService s ON m.visitID = s.visitID

INNER JOIN maintenanceAction i ON s.actionID = i.actionID

GROUP BY customerID;

10.Report on the steady customers based on the gross amount of money that we have gotten from them in the past year. Take into account any free oil changes that they have received, as well as any loyalty points that they may have used during that same time frame.

SELECT customerID, FROM vehicle

WHERE VIN =

(SELECT )

11. List the three premier customers who have paid Dave’s Automotive the greatest amount in the past year, and the sum of their payments over that period. Be sure to take into account any  
discounts that they have earned by referring prospective customers.

SELECT c.customerID, sum(actionCost) AS sum FROM premierCustomer c INNER JOIN vehicle v ON c.customerID = v.customerID

INNER JOIN maintenanceVisit m ON v.vin = m.vin

INNER JOIN visitService s ON m.visitID = s.visitID

INNER JOIN maintenanceAction i ON s.actionID = i.actionID

group by customerID order by sum DESC LIMIT 3

Haven’t included discount

12. List the five model, make, and year that have caused the most visits on average to Dave’s  
automotive per vehicle in the past three years, along with the average number of visits per  
Vehicle.

SELECT make, model, year, count(visitDate) as visitCount FROM premierCustomer c INNER JOIN vehicle v ON c.customerID = v.customerID

INNER JOIN maintenanceVisit m ON v.vin = m.vin

INNER JOIN visitService s ON m.visitID = s.visitID

INNER JOIN maintenanceAction i ON s.actionID = i.actionID

where visitDate >= '2014-12-08' AND visitDate <= '2017-12-08'

group by make, model, year

Work but dont have enough data !

13. Find the mechanic who is mentoring the most other mechanics. List the skills that the mechanic is passing along to the other mechanics.

SELECT mentorID, skillName FROM Mentorship

WHERE mentorID =

(SELECT mentorID FROM Mentorship

GROUP BY mentorID

ORDER BY mentorID ASC

LIMIT 1);

14. Find the three skills that have the fewest mechanics who have those skills.

SELECT skillName AS Skill, COUNT(skillName) FROM MechanicSkills

ORDER BY COUNT(skillName) ASC

LIMIT 3;

15. List the employees who are both service technicians as well as mechanics.

SELECT Employee.employeeID, lastName, firstName FROM Employee

WHERE Employee.employeeID IN

(SELECT Mechanic.employeeID FROM Mechanic

WHERE Mechanic.employeeID IN

(SELECT serviceTechnician.employeeID FROM serviceTechnician));

16. Three additional queries that demonstrate the five additional business rules. Feel free to create additional views to support these queries if you so desire.

16a. Rule 2: List the classified vehicle difficulties along with the mechanics that are qualified to service them

SELECT make, model, year, difficulty, lastName, firstName, YEAR(CURDATE()) - YEAR(dateHired) AS ExperienceYears

FROM vehicleRepairInterval

CROSS JOIN Mechanic INNER JOIN Employee ON Mechanic.employeeID = Employee.employeeID

WHERE

difficulty = 'easy' AND YEAR(CURDATE()) - YEAR(dateHired) >=1 OR

difficulty = 'medium' AND YEAR(CURDATE()) - YEAR(dateHired) >=2 OR

difficulty = 'hard' AND YEAR(CURDATE()) - YEAR(dateHired) >=3;

16b. Rule 3: Display the maintenance services along with the number of mechanics required to perform them.

SELECT serviceName, numOfRequiredMechanics FROM maintenanceItem

16c. Rule 4: Display each maintenance item and the required maintenance items to be performed before them.

SELECT m1.serviceName AS PrerequisiteService, m2.serviceName AS PostConditionService FROM maintenanceItem m1

INNER JOIN taskSequence t1 ON m1.actionID = t1.firstItem

INNER JOIN maintenanceItem m2 ON t1.nextItem =m2.actionID;

**--RESET VIEWS**

DROP VIEW Customer\_v;

DROP VIEW Customer\_addresses\_v;

DROP VIEW Mechanic\_mentor\_v;

DROP VIEW Premier\_profits\_v;

DROP VIEW Prospective\_resurrection\_v;

**--VIEWS**

1. Customer\_v – for each customer, indicate his or her name as well as the customer type (prospect, steady or premier) as well as the number of years that customer has been with us.

CREATE VIEW Customer\_v AS

(SELECT firstName, lastName, 'NA' AS CompanyName, 'PremierCustomer' AS CustomerType, YEAR(CURDATE()) - YEAR(dateSignedUP) AS YearsWithDave

FROM premierCustomer

INNER JOIN customer ON premierCustomer.customerID = customer.customerID

INNER JOIN individualCustomer ON premierCustomer.customerID = individualCustomer.customerID)

UNION

(SELECT 'NA', 'NA', coName, 'PremierCustomer' AS CustomerType, YEAR(CURDATE()) - YEAR(dateSignedUP) AS YearsWithDave

FROM premierCustomer

INNER JOIN customer ON premierCustomer.customerID = customer.customerID

INNER JOIN corporateCustomer ON premierCustomer.customerID = corporateCustomer.customerID)

UNION

(SELECT firstName, lastName,'NA', 'SteadyCustomer', YEAR(CURDATE()) - YEAR(dateSignedUP)

FROM steadyCustomer

INNER JOIN customer ON steadyCustomer.customerID = customer.customerID

INNER JOIN individualCustomer ON steadyCustomer.customerID = individualCustomer.customerID)

UNION

(SELECT 'NA', 'NA',coName, 'SteadyCustomer', YEAR(CURDATE()) - YEAR(dateSignedUP)

FROM steadyCustomer

INNER JOIN customer ON steadyCustomer.customerID = customer.customerID

INNER JOIN corporateCustomer ON steadyCustomer.customerID = corporateCustomer.customerID)

UNION

(SELECT firstName, lastName,'NA', 'ProspectiveCustomer', YEAR(CURDATE()) - YEAR(dateSignedUP)

FROM prospectiveCustomer

INNER JOIN customer ON prospectiveCustomer.customerID = customer.customerID

INNER JOIN individualCustomer ON prospectiveCustomer.customerID = individualCustomer.customerID)

UNION

(SELECT 'NA', 'NA',coName, 'ProspectiveCustomer', YEAR(CURDATE()) - YEAR(dateSignedUP)

FROM prospectiveCustomer

INNER JOIN customer ON prospectiveCustomer.customerID = customer.customerID

INNER JOIN corporateCustomer ON prospectiveCustomer.customerID = corporateCustomer.customerID);

2. Customer\_addresses\_v – for each customer, indicate whether they are an individual or a corporate account, and display all of the addresses that we are managing for that customer.

CREATE VIEW Customer\_addresses\_v AS

(SELECT i.lastName, i.firstName, 'N/A' as coName, 'IndividualCustomer' AS ContactType, 'Home' as AddressType, i.street, i.city, i.state, i.zipcode

FROM individualCustomer i

)

UNION

(SELECT 'N/A' as lastName, 'N/A' as firstName, c.coName, 'CorporateCustomer', a.type, a.street, a.city, a.state, a.zipcode

FROM corporateCustomer c

INNER JOIN addressLine a ON c.customerID = a.customerID

);

3. Mechanic\_mentor\_v – reports all of the mentor/mentee relationships at Dave’s, sorted by the name of the mentor, then the name of the mentee.

CREATE VIEW Mechanic\_mentor\_v AS

(SELECT a.lastName as MentorLastName, a.firstName as MentorFirstName, b.lastName as MenteeLastName,b.firstName as MenteeFirstName, m.skillName, m.startDate, m.endDate

FROM Employee a

INNER JOIN Mentorship m ON a.employeeID = m.mentorID

INNER JOIN Employee b ON b.employeeID = m.menteeID

ORDER BY a.lastName);

4. Premier\_profits\_v – On a year by year basis, show the premier customer’s outlay versus what they would have been charged for the services which they received had they merely been steady customers.

CREATE VIEW Premier\_profits\_v AS

SELECT customerID, () AS MoneySaved FROM

//shows the annual payment made by a premier customer

SELECT (monthlyInstallment\*12) AS AnnualPayment FROM premierCustomer

//shows the totalcost of vsits made by a premier customer

SELECT SUM(actionCost) FROM MaintenanceAction

INNER JOIN

(sum(totalcost)) - (monthlyInstallment\*12)

Yearly cost - total cost of all the things

-- not done

5. Prospective\_resurrection\_v – List all of the prospective customers who have had three or more contacts, and for whom the most recent contact was more than a year ago. They might be ripe for another attempt.

CREATE VIEW Prospective\_resurrection\_v AS

SELECT distinct customerID IN

(SELECT customerID FROM dateContacted

GROUP BY customerID

HAVING COUNT(customerID) >= 3),

dateContacted

FROM dateContacted

WHERE dateContacted < '2017'

ORDER BY dateContacted DESC;

**--DISPLAY VIEWS**

SELECT \* from Customer\_v;

SELECT \* from Customer\_addresses;

SELECT \* from Mechanic\_mentor\_v;

SELECT \* from Premier\_profits\_v ;

SELECT \* from Prospective\_resurrection\_v;